НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено:
Завідувач кафедри
_____ *Сергій СТІРЕНКО*
(підпис)
“___”_____20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інженерія програмного забезпечення комп’ютерних систем»

спеціальності 121 «Інженерія програмного забезпечення»

на тему: «Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення»

Виконала:
студентка _IV_курсу, групи__ІП-62__
Печена Марина Василівна _____
(підпис)

Керівник:
Доцент, кандидат технічних наук
Порєв Віктор Миколайович _____
(підпис)

Консультант з нормоконтролю:
Професор, доктор технічних наук
Сімоненко Валерій Павлович _____
(підпис)

Рецензент

_____
(підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМ. ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення комп'ютерних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ *Сергій СТІРЕНКО* (підпис)
"___"_____20__ р.

ЗАВДАННЯ

на дипломний проєкт студентці

Печеній Марині Василівні

1. Тема проєкту «Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення»
керівник проєкту Порєв Віктор Миколайович, доцент, кандидат технічних наук
затверджена наказом по університету від «___» ___ 2020р. №_____
2. Термін здачі студентом закінченого роботи: «___» ___ 2020р.
3. Вихідні дані до проєкту: технічна документація, теоретичні та статистичні дані.
4. Зміст пояснювальної записки: опис предметної області, огляд веб-сервісу та вибір інструментів розробки, розробка програмного забезпечення, аналіз розробленого програмного забезпечення.

5. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
нормоконтроль	д.т.н., проф. Сімоненко В. П.		

6. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітки
1.	Затвердження теми роботи	13.09.2019 - 10.02.2020	
2.	Вивчення та аналіз завдання	11.02.2020 - 13.04.2020	
3.	Написання вступної частини та огляд предметної області	13.04.2020 - 20.04.2020	
4.	Розбір принципу роботи, аналіз переваг та недоліків системи.	20.04.2020 - 4.05.2020	
5.	Дизайн системи та розробка алгоритму роботи	4.05.2020 - 14.05.2020	
6.	Програмна реалізація системи	14.05.2020 – 24.05.2020	
7.	Оформлення документації дипломної роботи	24.05.2020 - 31.05.2020	
8.	Передзахист		
9.	Захист		

Студент _____ Марина ПЕЧЕНА
(підпис)

Керівник проєкту _____ Віктор ПОРЄВ
(підпис)

АНОТАЦІЯ

В даному бакалаврському дипломному проекті реалізовано веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення. Веб-сервіс надає користувачу можливість реєстрації, взаємодії із профілем та особистим кабінетом, створення/редагування нейромереж та їх навчання. Також сервіс дає змогу візуалізувати результати навчання нейромереж первної групи на графіку у вигляді кривих з різними проектами по їхній спільній групі.

Програмний продукт був створений за допомогою HTML та CSS, з використанням мови Javascript у середовищі редагування тексту Sublime, та браузері. Веб-сервіс розрахований для використання як на комп’ютері, так і на телефоні чи планшеті.

Дизайн даного веб-сервісу був побудована на основі вже існуючих багатьох сайтів-аналогів з власним кабінетом, взаємодією із файлами та програмним кодом. Враховані усі їхні переваги недоліки, а саме дизайн, налаштування та оформлення зовнішнього вигляду.

Розмір пояснювальної записки – 87 аркушів, містить 78 ілюстрацій, 4 додатки

SUMMARY

This bachelor's degree project implements a web service with a microservice architecture for modeling wide-purpose neural networks. The web service provides the user with the ability to register, interact with the profile and personal account, create / edit neural networks, and train them. The service also allows you to visualize the results of neural networks' training of the original group on a graph in the form of curves with various projects for their general group.

The software product was created with HTML and CSS, using Javascript in the Sublime text editing environment, and the browser. The web service is designed for use on a computer, as well as on a phone or tablet.

The design of this web service was built on the basis of many already existing similar sites with their own account, file interaction and software code. All their advantages and disadvantages are taken into account, including design, settings and interface.

The size of the explanatory note – 87 sheets, contains 78 illustrations, 4 adds.

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

до дипломної роботи
освітньо-кваліфікаційного рівня бакалавр

на тему: “ Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення ”

Київ – 2020 року

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проект	3	
2	А4	ІАЛЦ.467100.001 ВП	Відомість дипломного проєкту	1	
3	А4	ІАЛЦ.467100.002 ТЗ	Технічне завдання	3	
4	А4	ІАЛЦ.467100.003 ПЗ	Пояснювальна записка	87	
5	А3	ІАЛЦ.467100.004 Д1	Блок-схема залежності об’єктів фронтенда	1	
6	А3	ІАЛЦ.467100.005 Д2	Ілюстрація взаємодії послідовності подій та їх обробка	1	
7	А4	ІАЛЦ.467100.006 Д3	Схема взаємодії модулів проєкта	1	
8	А4	ІАЛЦ.467100.007 Д4	Лістинг	55	

					ІАЛЦ.467100.001 ВП		
Змн.	*Арк.*	*№ докум.*	*Підпис*	*Дата*			
Розроб.		*Печена М.П.*			*Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення* *Відомість дипломного проекту*	*Літ.*	*Арк.*	*Аркушів*
Перевір.		*Порєв В.М.*					*1*	*1*
						НТУУ «КПІ ім. Ігоря Сікорського», ФІОТ ІП-62		
Н. Контр.		*Сімоненко В.П.*						
Затверд.								

ТЕХНІЧНЕ ЗАВДАННЯ

до дипломної роботи
освітньо-кваліфікаційного рівня___бакалавр___

на тему: “ Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення ”

Київ – 2020 року

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Дане технічне завдання поширюється на розробку програмного продукту по темі «Веб сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення». Область застосування: навчання в рамках тем пов’язаних із нейромережами.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на виконання роботи кваліфікаційно-освітнього рівня «бакалавр програмної інженерії», затверджене кафедрою обчислювальної техніки Національного технічного Університету України «Київський Політехнічний інститут ім. Ігоря Сікорського»

3. МЕТА І ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою даного проекту є розробка веб-сервісу із мікросервісною архітектурою для моделювання нейромереж широкого призначення.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом розробки є науково-технічна література з теорії і практики програмування, публікації в Інтернеті з даних питань.

5. ТЕХНІЧНІ ВИМОГИ

5.1. ВИМОГИ ДО РОЗРОБЛЯЄМОГО ПРОДУКТУ

- Зручний програмний інтерфейс.
- Доступ до власного кабінету, можливість зберігати проекти, редагувати їх, та переглядати інші за допомогою пошуку.
- Можливість створення, редагування, навчання та перегляду

нейромереж. Відображення результату навчання у вигляді графіку із кривими відповідно до проектів у групі.

- Зручність та користь використання веб-сервісу, що включає в себе можливість авторизації, збереження проектів, пошуку, додавання у обране, можливість переглянути навчання нейромережі та статистику.
- Технічна коректність виконання усіх заданих алгоритмів.

5.2. ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

- Пристрій із встановленим браузером.
- будь-яке середовище розробки веб-сторінок або текстовий редактор

5.3. ВИМОГИ ДО АПАРАТНОЇ ЧАСТИНИ

- Доступ до мережі Інтернет.

6. ЕТАПИ РОЗРОБКИ

	Дата
Затвердження теми роботи	13.09.2019 - 10.02.2020
Вивчення та аналіз завдання	11.02.2020 - 13.04.2020
Написання вступної частини та огляд предметної області	13.04.2020 - 20.04.2020
Розбір принципу роботи, аналіз переваг та недоліків системи.	20.04.2020 - 4.05.2020
Дизайн системи та розробка алгоритму роботи	4.05.2020 - 14.05.2020
Програмна реалізація системи	14.05.2020 – 24.05.2020
Оформлення документації дипломної роботи	24.05.2020 - 31.05.2020
Передзахист	« » _____ 2020
Захист	« » _____ 2020

Пояснювальна записка
до дипломного проекту

на тему: «Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення»

Київ – 2020

ЗМІСТ

СПИСОК СКОРОЧЕНЬ

МСА – мікросервісна архітектура

МС – мікросервіс

БД – база даних

ОК – онлайн комерція

МП – мікропрограма

ф-енд – фронтенд

б-енд – бекенд

ВСТУП

Даний дипломний проэкт призначений для огляду можливих рішень розробки веб-сервісу з елементами мікросервісу, та створення свого для моделювання нейромереж широкого призначення.

Технології у сучасному світі розвиваються дуже швидко, а з ними і функціонал програмування. Все більше і більше програм повинні мати в собі не лише хороший код, а ще й відповідати потребам людей. Тому розмір популярних програм постійно зростає, доповнюючись необхідним функціоналом, або розширяючи поточний. Через це актуальним є питання розробки програм з такою архітектурою, що дозволить легко і швидко вносити зміни в програму, що вже існує, або додавати нові фрагменти коду. При цьому дуже важливо, щоб навіть при внесенні змін програма продовжувала працювати. На сьогоднішній день багато зусиль у розробці великих програм спрямовано на те, щоб серверна частина знаходилася на великій та потужній машині, з якої в серверній здуватимуть пилинки. Однак люди часто упускають з виду простіші рішення проблем. Таким рішенням і є реалізація мікросервісної архітектури. Вона дозволить не лише змінювати частину коду, не змушуючи інші частини зупинитися, а ще й використовувати замість однієї громіздкої дорогої машини декілька менших та дешевших. До того ж, актуальним є питання універсальності написаних програмних рішень та засобів їх розробки. Найкращим варіантом з точки зору зручності розробки є варіант, коли розробники пишуть свою частину на тій мові та у тому середовищі, яке добре знають, та додають свої рішення в готову програму, не чекаючи тих, хто працює повільніше, або виконує певну більш громіздку задачу.

РОЗДІЛ 1. ОГЛЯД ЗАГАЛЬНИХ ВІДОМОСТЕЙ ПРО МІКРОСЕРВІСИ

1.1. Що нам відомо про мікросервіси

Мікросервіси (далі МС) – це незалежні один від одного блоки коду, на які розбита програма. Вони можуть працювати окремо, розроблятися різними мовами програмування, але всі об’єднані тим, що разом виконують глобальну поставлену задачу. [1]

Мікросервісна архітектура (МСА) – це такий підхід до створення програми, при якому один блок коду розбивається на інші, менші за розміром – МС, кожен із яких працює в окремому процесі. Всі разом вони утворюють готову програму або вирішують глобальну задачу.

МС можуть виконуватися, деплоїтися (тобто запускатися в необхідному середовищі) та тестуватися незалежно один від одного. Більше того, МС можуть знаходитися на різних серверах, операційних системах, або навіть бути написаними різними мовами програмування.

1.2. Моноліт і чому з’явилися МС

З архітектурою розібралися швидко, тепер зупинімося детальніше на мікросервісах. Зрозуміти їх суть легше всього якщо піти від супротивного – моноліту. Саме у порівнянні з ним, або навіть протиставленні, можна все побачити. Отож, почнемо з визначення.

Моноліт – це чимала програма, написана одним шматком коду. При цьому, не виключається її модульність. Мається на увазі що при запуску якогось уривку залучено увесь код.

Мені подобаються прості приклади, тому я вважаю що далі цю тему можна розглянути через призму кошика фруктів. Так можна просто і легко зрозуміти основну думку.

Рисунок 1.1. Монолітна архітектура

Будування програм, використовуючи монолітний підхід – досить очевидний спосіб, та перше що приходить на думку при їх реалізації.

Єдиний процес – є логікою такої програми. При цьому ви можете також використовувати можливості обраної вами мови програмування для розділення програми на класи, функції та простори імен, однак будь-які зміни у коді потребують нового збору всього моноліту.

Такі програми можуть бути успішними, однак все більше і більше людей розчаровуються у них. З плином часу зберігати хорошу модульну структуру стає дедалі складніше, а зміни логіки одного модуля призводять до зміни коду інших.

Ці незручності викликали появу МСА, зміст якої у побудові програми у вигляді набору сервісів.

Рисунок 1.2. Мікросервісна архітектура

Ці незручності викликали появу МСА, зміст якої у побудові програми у вигляді набору сервісів. Така архітектура дозволяє використовувати кожну одиницю функціональності в окремому сервісі.

Мікросервіси не є панацеєю, і нема потреби постійно будувати програму на них. Однак у випадку якщо нам знадобиться масштабувати програму, то з монолітом користувач, як і розробник, отримає більше проблем, ніж переваг.

Повернемося до прикладу із фруктами, і тепер нам необхідно приготувати із них компот. Якщо ми купуємо наборами по 4 фрукти то це досить не зручно, коли потрібно зробити напій без одного фрукта, або з подвійною кількістю іншого. Однак якщо купувати окремо кожен вид фруктів, то можна готувати те що потрібно в яких завгодно пропорціях та комбінаціях. Це і є масштабування програмного продукту.

Наочно це можна побачити в наступних рисунках:

Рисунок 1.3. Масштабування моноліту

Рисунок 1.4. Масштабування МС

Тож виділимо 6 основних властивостей МС:

- невеликий розмір
- незалежність
- відповідає за одну функціональну можливість
- складається з одного або декількох процесів
- наявність власної бази даних
- автоматизація процесів розробки та підтримки

1.3. Переваги та недоліки МС

1.3.1. Невеликий розмір

Що ж означає «невеликий» і чому не можна сказати інакше?

Мається на увазі не кількість рядків коду та не розмір МС. Все це залежить від складності та функціональних можливостей. Кожен повинен самостійно визначитися з розміром, що краще всього робиться на практиці. Однак, розмір МС повинен бути таким, щоб виконувалася одна із наступних умов:

1. Один сервіс може розробляти команда не більше ніж із 12 чоловік.

   У лідерів проекту є правило щодо розміру команди: він повинен бути таким, щоб всіх можна було нагодувати 2-ма піцами (two-pizza-team, що в перекладі з англійської означає «команда на 2 піци»)
2. Команда із 6 людей може супроводжувати 6 сервісів

   Під «супроводжувати» маються на увазі всі аспекти підтримки робото спроможності МС та можливість його використати. Сюди відноситься і розробка нового функціоналу, і виділення нових МС із розрісшихся старих, моніторинг, тестування тощо.
3. Контекст одного сервісу розміщується у голові в однієї людини

4. Замінність

   МС повинен бути таким, щоб його можна було переписати з нуля за розумний період часу. Іншими словами, команда, що супроводжує МС, повинна бути в змозі змінити поточну реалізацію на нову, при чому в звичайному робочому темпі. Ще іншими словами, один сервіс може бути повністю переписаний однією командою за одну Agile-ітерацію.

1.3.2. Незалежність

У контексті МС незалежність означає, що їх можна розгортати окремо один від одного. Тобто, при зміні МС повинна бути можливість ввести його в експлуатацію, не чіпаючи будь-які інші частини системи. Інші МС у цій системі повинні виконуватися і працювати під час змін у якомусь МС, і після того як він оновиться.

Розглянемо вже готовий приклад

Рисунок 1.5. Розгортання програми з МС

Під час розгортання МС Orange, бачимо що інші МС продовжують працювати та обслуговувати користувачів/виконувати розрахунки.

Можливість індивідуального розгортання кожного із МС важлива, бо в системі їх може бути дуже багато, і кожен може взаємодіяти з деякими іншими. Одночасно із цим виконується розгортання деяких (або навіть усіх

МС), і якби довелося розгортати всіх у строгому порядку, то ця задача стала би дуже громіздкою, і в результаті привела б до нечастих, об'ємних та ризикованих розгортань. Тому перевага надається саме МС, із можливістю частих та безпечних розгортань.

Однак, щоб мати таку можливість, необхідно врахувати наступне при процесі побудови програми:

1. Всі МС вбудовані в окремі артефакти або пакети.
2. Процес розгортання забезпечує підтримку індивідуального розгортання при функціонуванні інших МС. Наприклад, процес плавучого розгортання, при якому МС розгортається на одному сервері за один раз, щоб зменшити час простою.

Цей підхід до створення МС змінює спосіб їхньої взаємодії. Тепер кожен МС повинен бути стійким до виникання помилок у інших, і по можливості працювати далі самостійно. Збій одного МС, наприклад під час простою, не повинен призводити до збою інших, а лише до скорочення функціональності, або більшому часу обробки.

1.3.3. Відповідає за одну потребу

Визначення меж МС – найважливіший крок. Від цього буде залежати його майбутнє життя, і це серйозно відображається на житті команди, що відповідає за нього.

Основним принципом визначення зони відповідальності МС – сформувати її навколо бізнес-потреби. Один МС відповідає лише за одну функціональну потребу, тобто:

1. На МС лежить лише один обов'язок
2. Цей обов'язок у реалізації окремої потреби.

Нагадує відомий принцип єдиного обов'язку, а саме «У класу має бути лише одна причина для зміни». Хоча мова тут іде про класи, цей принцип можна застосовувати поза межами класу, так, у випадку МС, він

застосовується на рівні сервісів.

МС повинен реалізовувати лише одну функціональну потребу, та змінюватися при зміні цієї потреби.

Існує 2 види потреб у системі МС:

1. **Бізнес-потреба** – виконана системою дія, що допомагає реалізації задачі системи.

   Наприклад, відстежування кошика покупців, або підрахунок ціни.

2. **Технічна потреба** – дещо необхідне для деяких інших МС, по типу інтеграції з якоюсь сторонньою системою

   Технічні потреби не можна назвати головними рушійними силами при розбитті системи на МС, вони виникають в основному тоді, коди виявляється, що декільком МС для її бізнес-потреб необхідна одна і та ж технічна можливість.

Чим компактніша зона відповідальності МС, навколо якої він побудований, тим більш формалізовані її взаємозв’язки з іншими областями, і тип легше створювати новий МС.

Коли межі задані, а він виділений в окрему кодову базу, захистити ці межі від зовнішнього впливу не складно. Просто варто далі працювати всередині самого МС, створивши мікросвіт, використавши патерн «Обмеженого контексту» (Bounded Context). У МС для будь-якого об’єкту та дії може бути своя інтерпретація, відмінна від інших контекстів.

Розглянемо тепер аналогію обмеженого контексту. Якщо чотирьох новонароджених черв’ячків розмістити в різних фруктах, то у кожного з них вже буде своя версія навколишнього середовища, тобто поняття м’якоті, кісточок, положення всередині-ззовні для них залишилися однаковими, але для кожного воно існує в рамках свого особистого фрукта.

Рисунок 1.6. Обмежений контекст кожного МС

Але що ж робити якщо межі виявилися неправильними? У такому випадку зміна функціональності одного МС призведе до зміни і інших. В результаті можуть зіпсуватися інтерфейси всіх залежних МС, а також тестів. А якщо ці МС ще й належать різним командам, то доведеться проводити між командні зустрічі, шукати вільний час, і т.д. Тому правильні межі МС – основа здорової МСА.

1.3.4. Складається з одного або декількох процесів

МС повинен виконуватися в окремому процесі (або окремих процесах), щоб залишатися як можна довше незалежним від інших МС системи, а також щоб зберігати можливість самостійного розгортання.

Тобто, маємо:

1. Усі МС працюють в окремих від інших МС процесах
2. У будь-якого МС може бути більше одного процесу.

Повернемося до нашого прикладу. Тепер нехай авокадо та слива

символізують собою мікросервіси Plum та Avocado відповідно. Якщо МС Plum та Avocado працюватимуть в одному процесі, то код МС Plum може викликати побічні ефекти в МС Avocado. Це означає сильну та небажану залежність між ними. Робота одного може призвести до простою або помилки іншого.

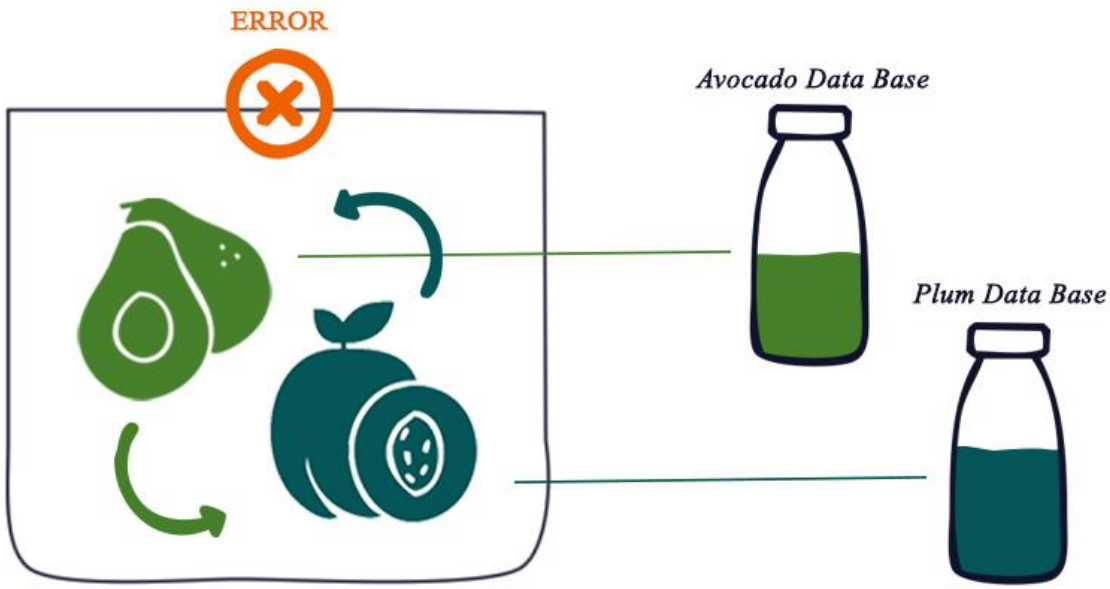


Рисунок 1.7. Декілька МС у одному процесі

При виконанні декількох МС в одному процесі може виникнути помилка на його межі.

Тобто, припустимо що при розгортуванні нової версії Plum, виникне потреба у повторному розгортанні Avocado, або підтягуванні динамічного завантаження коду для зміни Plum у процесі роботи. Перший варіант протилежний до принципу про індивідуальне розгортання, а другий складний і ставить процеси під загрозу збою.

Так чому ж МС може складатися із більш ніж одного процесу якщо ми прагнемо максимально його спростити? Це легко зрозуміти. Оскільки є такі процеси, які виконують певні алгоритми, а також використовують бази даних, необхідні для них. Алгоритми виконуватимуться в одному процесі, а взаємодія з БД відбуватиметься в іншому.

Часто МС потребує 2 або більше процесів для реалізації необхідних

системі відповідних функціональних потреб. Наприклад, сховище даних, та їх фонова обробка.

1.3.5. Наявність власної Бази Даних

Ще один із важливих елементів у парадигмі МС: Кожному мікросервісу по своїй базі даних (далі БД).

Це ще один наслідок того, що кожен МС працює із завершеною функціональною потребою. Великій частині бізнес-можливостей необхідне якесь сховище даних.

Насправді в моноліті теж можна поборотися за відокремленість компонентів, на рівні серверного коду. Якщо відокремленість перестала бути ізольованою, то доводитиметься рефарктурити код. Але через 2-3 роки в БД настане такий хаос, що рефакторінг навряд чи зможе допомогти. Доведеться витрачати безліч часу та ресурсів щоб виправити помилку, якої можна було уникнути.

У МСА проблема спільної БД вирішується дуже просто – Спільної БД просто немає.

Наприклад, процес Avocado зберігає інформацію про свої елементи. Тож тепер слабка залежність цього МС з іншими полягає в тому, що це сховище даних належить повністю йому. Лише він визначає як і коли зберігається інформація. Інші ж МС, як показано на рисунку, можуть звертатися до цієї інформації лише через доступ у інтерфейс Avocado, але не напряму у його сховище даних (рис. 1.3.4).

Такий підхід дозволяє кожному МС використовувати оптимальну для його задач БД, що дає переваги у часі розробки, продуктивності та масштабуванні. Очевидний недолік – необхідність у адмініструванні та веденні декількох баз даних, а також робота з ними.

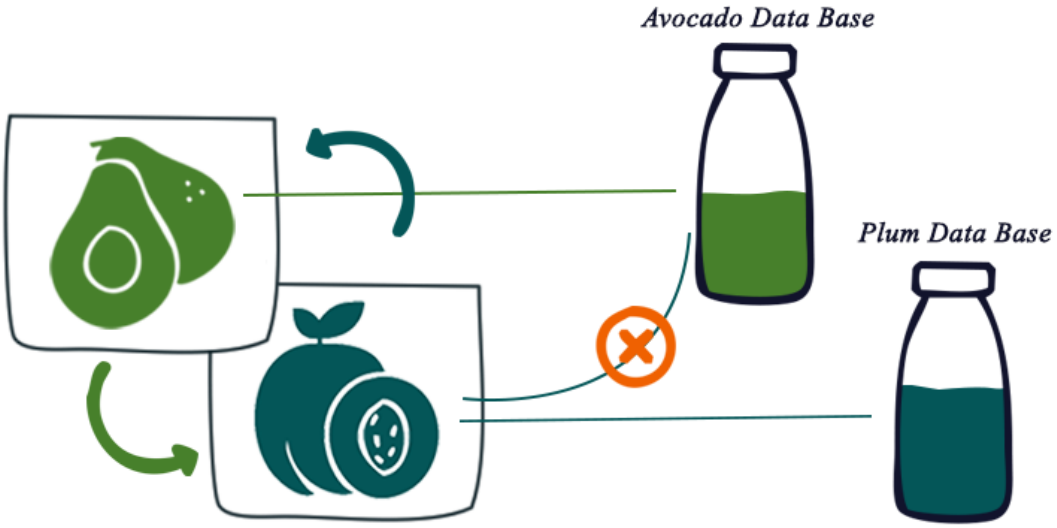


Рисунок 1.8. Звернення МС до БД

1.3.6. Автоматизація процесів розробки та підтримки

Одне із найбільш критичних місць у МСА – це необхідність розробляти код для розподіленої системи, елементи якої взаємодіють через мережу.

А мережа не надійна сама по собі. Вона може просто відмовити, може погано працювати, може раптом перестати пропускати який-небудь тип повідомлень, бо змінилися налаштування файервола. Десятки причин та видів недоступності.

Тому МС повинні враховувати можливість того, що вони можуть іноді перестати відповідати, а іноді відповідати повільніше. Та повинні вміти коректно відновлювати свою роботу, обробляти різні види відмов, вміти чекати.

Тож зазвичай проблему такої складності вирішують так:

1. Не доводять систему до ідеального стану.

   По-перше, це дуже дорого.

   Звичайно, це не означає що система погана, або рушиться при першому випробуванні. Вона просто відповідає необхідним критеріям, виконує необхідний функціонал, але у ній можуть бути присутні помилки, що мають незначний вплив на її стійкість та продуктивність

2. З іншого боку роблять вклад у інфраструктуру, яка допомагає швидше виключати непередбачувані ситуації

   Необхідно покривати тестами значну частину коду. Повинен бути інтелектуальний моніторинг, який не лише моментально показує неробочі місця, а й сигналізує про погіршення стану системи з прогнозуванням можливих збоїв

ВИСНОВКИ ДО РОЗДІЛУ 1

У попередньому розділі було переглянуто суть МС, їх недоліки та переваги у використанні. Так чим же саме мікросервіси краще моноліту?

Архітектурний стиль МС враховує необхідність безперервного доставлення ПО. Це робиться завдяки акценту на сервісах, які:

- Можна швидко розробити та змінити
- Можна тестувати з усіх боків за допомогою автоматизованих тестів
- Можна розгортати незалежно один від одного
- Працюють ефективно
- Ці властивості мають ряд недоліків з якими доведеться зустрітися.
- Наприклад, організаційні питання:
- Як утримувати в узгодженні версій сотню мікросервісів, які ще й постійно непередбачуваного розгортаються.
- Як забезпечити доступ до середовища у кожного інженера кожної команди?
- Яка команда писатиме інтеграційні тести? А якщо хтось і погодиться то буде нелегко написати їх для такої складної конфігурації
- А якщо виникає помилка то чия вона?

Так, це проблеми реалізації. На щастя, їх можна вирішити практикуючи Agile та DevOps.

Окрім організаційних може виникнути ряд архітектурних проблем, таких як розбиття єдиного синхронного моноліту на МСА, засновану на подіях, та на безліч маленьких блоків, у кожному з яких необхідно враховувати помилки.

Тоді ж навіщо розбивати моноліт на МС?

Якщо у вашому моноліті все ідеально, то розбивати його і не треба. Але якщо виникають проблеми, то варто подивитись, може МСА зможе вирішити їх.

Розбиття на незалежні компоненти дає безумовні переваги: легке розуміння контексту, гнучкість розвитку, управління та масштабування.

Незалежність та невеликий розмір кожного компонента дають неочікувані плюси з точки зору інфраструктури.

Також тепер нема потреби купувати могутню машину за великі гроші, адже МС можна встановити на звичайні дешеві машинки. А в результаті виявиться, що навіть разом вони коштуватимуть дешевше, але працювати ефективніше той супермашини яка була б необхідна при моноліті.

Перейдемо до перегляду реальних прикладів. Такі популярні сервіси як Amazon, Netflix, Google та інші показують хороші результати. Їх гнучкість, швидкість виводу нових продуктів вражають. Про це ми поговоримо у наступному розділі.

Але важливо пам’ятати що в таких організаціях багато чудових спеціалістів, тому їм складнощі МС не заважають.

РОЗДІЛ 2. АНАЛІЗ ІСНУЮЧИХ ВЕБ-СЕРВІСІВ ІЗ МСА

2.1. МС як елемент покращення роботи веб-сервісу

Команди, які розробляють МС надають перевагу незвичному методу стандартизації. Замість того, щоб використовувати набір попередньо заданих стандартів, написаних кимось, вони реалізовують ідею побудови корисних інструментів, які зможуть бути використаними іншими в ході розв'язання схожих проблем. Тепер, коли git та github стали стандартною системою контролю версій, опен-сорсні практики проводяться все частіше і частіше.

Для початку потрібно зрозуміти що таке онлайн комерція(далі ОК) і чому ми взагалі про неї заговорили. На щастя, тут все просто. ОК – (з англійської «E-commerce», або ще «електронна комерція») це підприємницька діяльність, яка пов’язана з рекламуванням, поширенням, продвиганням, продажем послуг чи товарів у мережі інтернету[6]. Тобто будь-які дії з комерційним відтінком у інтернеті вважаються ОК.

Тобто, кажучи простіше, ОК містить в собі наступні категорії: онлайн-продажі, інтернет-банкінги, бронювання квитків та готелів, транзакції в системах платіжки, онлайн-маркетинг, реклама.

З технічної точки зору ОК спирається на такі основні основи: сервер, БД, та система доставки товару чи послуги покупцю. Найважливіший компонент – це саме перша частина, якісний сервер. Великі компанії потребують роботи з БД, а доставляти електронні товари чи послуги – вже буде не складно.

Прикладом ОК є онлайн-сервіси прийому комунальних платежів, інтернет-магазини, сайти-каталоги, дошки оголошень, сайти які надають користувачам медіа-матеріали. Зазвичай робота відбувається по інтуїтивно зрозумілій схемі:

1. Користувач переглядає каталог та обирає продукт
2. Сервер отримує заявку на продукт та відправляє в систему обробки замовлень, де перевіряється наявність
3. Взаємодія із фінансовою системою для виводу кінцевої інформації і суми користувачу
4. Перелік коштів, підтвердження системою, відправлення заявки на склад, доставлення користувачу.

Оскільки мова йде про безліч незалежних операцій, то тут найнеобхідніше, щоб при виході однієї частини програми із ладу, інші продовжували працювати. Як цього досягти? МС – чудове рішення саме для великих і складних IT-систем. Тому багато компаній-гігантів використовують саме їх.

Архітектура послуг – це методика розробки програми, побудованої як набір маленьких сервісиків (мікропослуг), де кожен із них відповідає за одну функцію, як пошук продукту, підрахунок суми, оплата, взаємодіючі через шлюзи API.

10 років назад знайти приклад використання МСА було б дуже складно, навіть майже неможливо, але по мірі розвитку МС, росте і кількість компаній, що впроваджують їх у свою архітектуру.

Рисунок 2.1. Компанії, що використовують МСА

Після того як Amazon і Netflix пройшли через перехід від моноліту до МС, процес та суть МС прояснилися та стали доступними для використання їх послідовниками.

Одною із найбільших переваг МС для ОК є можливість розробки, масштабування та розгортання кожного сервісу окремо. Тобто, на прикладі інтернет-магазину, за необхідності вносяться зміни лише в мікропрограмі яка за це відповідає, не чіпаючи роботу основного сайту. Так МСА скорочує час на реліз фіч, виправлення помилок та виводу на ринок оновленого продукту.

Тож перейдемо до прикладів компаній, що пройшли процес перетворення моноліту на МС.

2.2. Amazon

Випадок Amazon – перший, де МС відіграли велику роль у зміні бізнесу. «Якщо повернутися у 2001 рік, то роздрібний сайт Amazon був великим архітектурним монолітом» – каже Роб Брігам, старший менеджер Amazon AWS по управлінню продуктами[1].

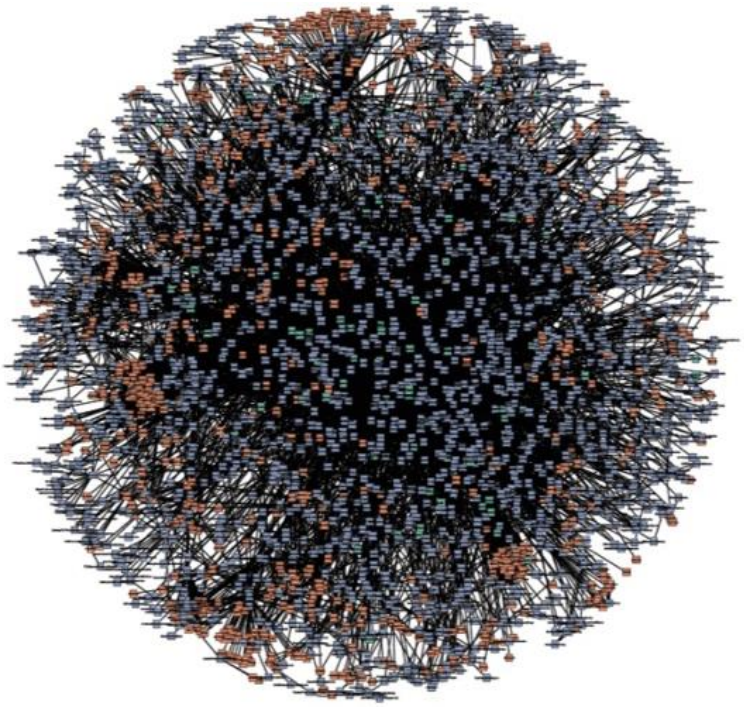


Рисунок 2.2. Структура МС Amazon-а, схожа на Зірку Смерті, але більш могутня[1]

Архітектура Амазона не була найбільшим монолітом у світі, однак через те, що вони мали сотні розробників, розподілених по організації, і вони втратили можливість оперативно вирішувати помилки.

Та навіть маючи стільки хороших розробників компанія застряла на тиждень при впровадженні цього. Розбивка великого шматка на маленькі програми дозволило їм побачити де саме сервіс просідає, де падає швидкість, та яка природа всього цього. Також дала можливість розробникам перебудувати сервіс орієнтуючись на МСА, даючи кожному сервісу по команді, яка повністю присвячувала себе йому.

Те, що починалося як очищення системи закінчилось еволюцією світу сучасної архітектури. Амазон після змін проклала шлях іншим компаніям, коли виклала розв'язання проблеми у відкритому доступі.[1]

2.3. BestBuy

Коли BestBuy займав десяте місце у рейтингу Internet Retailer Top 500, компанія дійшла до такої точки, що вносити зміни, навіть прості, та оновлювати систему стало не те що складно, а небезпечно. Тому після того як вони давно простояли на місці, в 2010 році було прийнято сміливе рішення.

Вони почали з побудови графіків взаємопов’язаних модулів в рамках моноліту та їх залежностей, після чого впровадили МС, крок за кроком використовуючи патерн Strangler для трансформації.

Не зважаючи на те, що перехід був складним, компанія використала його для перепроектування організації й системи для відповідності один одному і підтримки майбутніх процесів компанії.[2]

2.4. Coca Cola

Компанія Кока-Кола із 3800 продуктами по всьому світу та дочірніх компаніях ледь не в кожній країні світу, зіштовхнулася із проблемою об’єднання підприємств.[3] Оскільки багато з них знаходилися на інших

континентах, то їх об’єднання було складною задачею, як і підтримка зросту. Група програмістів прийняла рішення використати МС для цих цілей, а також поступово замінити старе програмне забезпечення. У цьому випадку швидкі зміни були неможливі через глобальне використання продукції.

Компанія обрала для переходу архітектуру з використанням моделі DevOps. В її рамках і впровадили структуру, поділену на незалежні програмки, які вирішували ряд питань.[3]

Наряду зі змінами у технологіях та архітектурі, Coca Cola сфокусувалась на процесах і людях, отримуючи достойний результат.

2.5. Netflix

Компанія Netflix починала з продажу та оренди DVD-дисків, і досягла популярності після переходу на МСА на AWS.[4]

Чому вони перейшли на МС?

Ще в серпні 2008 року в компанії відбувся серйозний збій через пошкодження БД, яке не дозволяло їм протягом трьох днів відгружати диски клієнтам. Після цього вони прийняли рішення відійти від єдиної точки відмови, яка давала змогу масштабуватися лише по вертикалі. Компанія перейшла до компонентів, які дозволяли масштабуватися по горизонталі, і до того ж були доступними.

Компанія відмовилася від приватних центрів обробки даних та перейшла до загальнодоступного хмарного сховища – AWS. Це і забезпечило горизонтальне масштабування. Та вони не просто перенесли свої системи на хмару, а перепрофілювали.[4]

Також вони час від часу зупиняють свої сервери у випадковому порядку, і слідкують щоб це не вплинуло на роботу з клієнтами.

Компанія Netflix є одним із перших впроваджувачів МС, і одним із найбільш обговорюваних. Їх перехід відбувався так:

Спочатку було перенесено закодовані фільми та інші програми, що не

мали вплив на користувача. Потім вони відділили елементи, звернені до клієнтів, такі як підпис акаунту, вибір фільму, вибір та налаштування пристрою. Від початку до завершення компанії було необхідно 2 роки, щоб розділити моноліт на МС, і в 2011 вони остаточно це зробили.

Сьогодні сервіс Netflix використовує більше ніж 500 МС і шлюзи API, які щоденно обробляють понад два мільярди запитів.

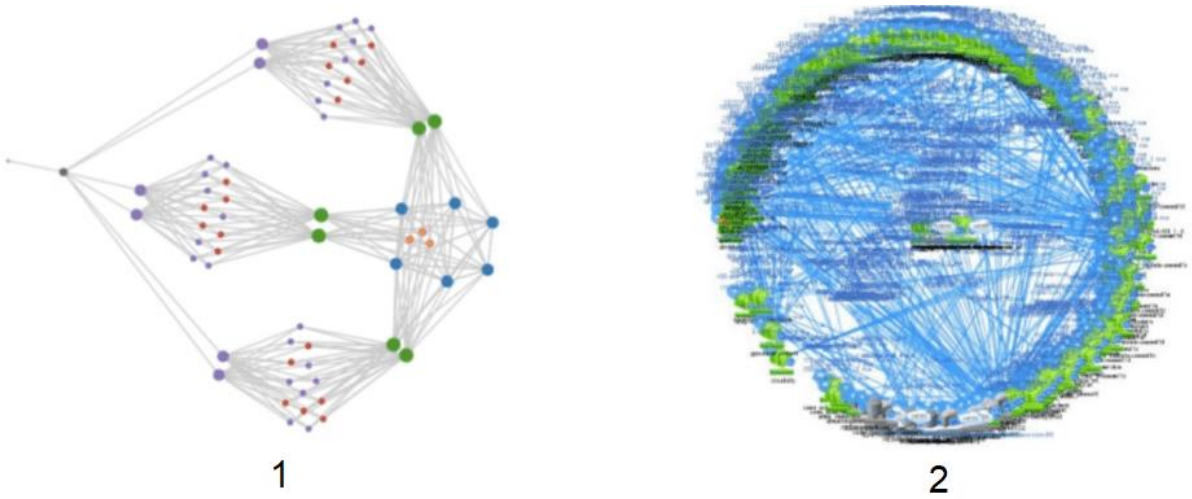


Рисунок 2.3. Архітектура Netflix. Проста(1) та звичайна(2) схеми [4]

2.6. Uber

В перші роки компанії Uber, коли вони лише вийшли на ринок, то реалізували досить простий монолітний сервіс. Перед ними стояла задача зробити всього декілька функцій, а саме: підключення водіїв та пасажирів, виставлення рахунку та оплата. Все це було реалізовано у межах одного міста, тому не було сенсу робити щось складніше за звичайний моноліт.

Згодом, коли сервіс почав розширятися та працювати з іншими містами, то почали і впроваджуватися нові послуги. В такому випадку програма стала стрімко зростати, і тоді розробники зрозуміли що треба щось змінювати. Основними недоліками системи була необхідність розгортати відразу весь проєкт при маленьких змінах, та велика відповідальність, яка змушувала лише давніх програмістів працювати із внесенням змін.

Тому в Uber почали розбивати один моноліт на декілька МС. Це

допомогло їм розв'язати всі проблеми в процесі, а також організувати безперервне постачання та прискорити швидкість росту бізнеса.

В рамках нової архітектури компанія реалізувала шлюз API та незалежні сервіси, які мають індивідуальні функції та можуть розгортатися незалежно один від одного.

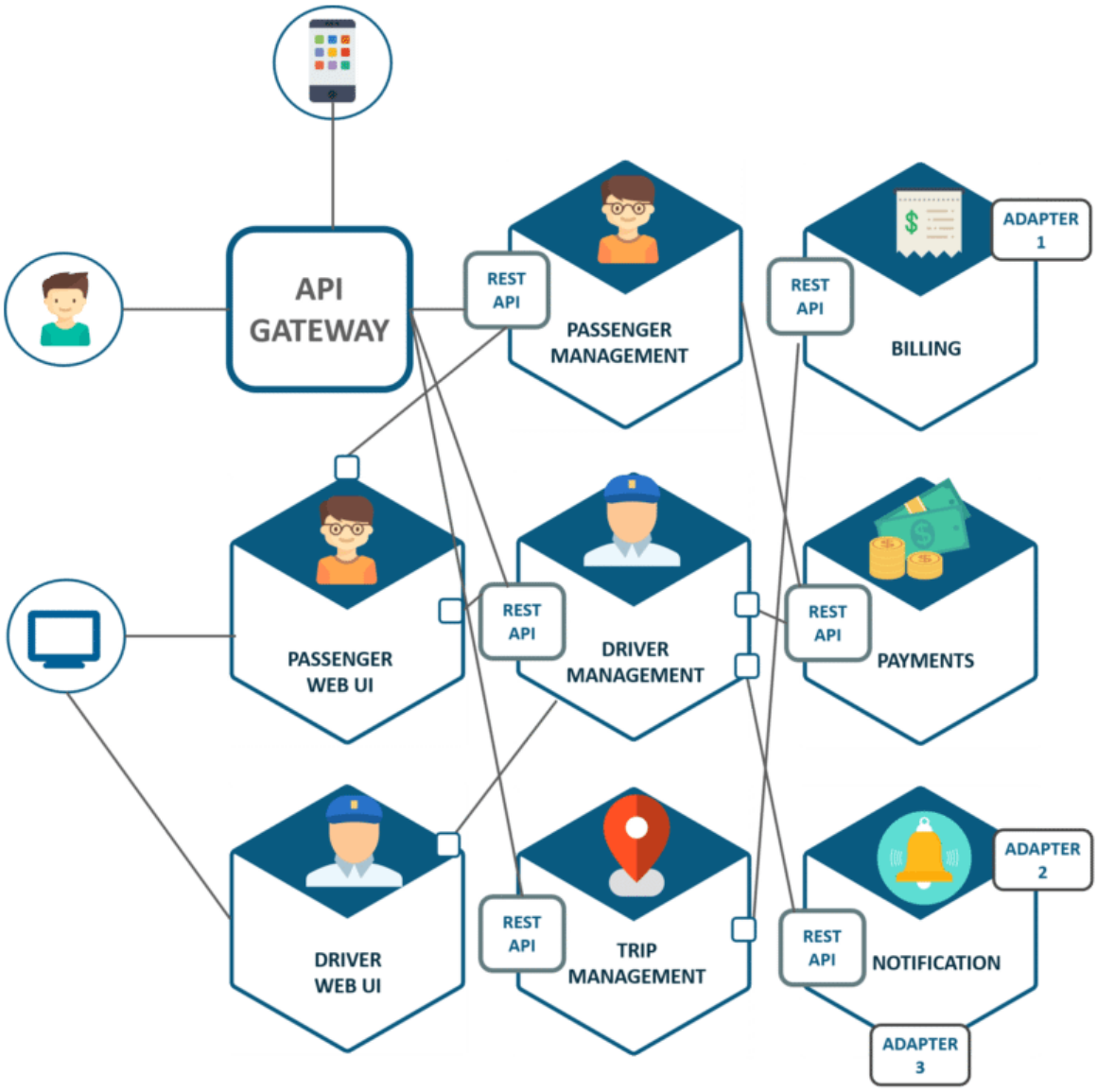


Рисунок 2.4. Модель MCA в компанії Uber[5]

ВИСНОВКИ ДО РОЗДІЛУ 2

Проаналізувавши приклади компаній, що успішно впровадили МСА у свої проєкти, приходимо до висновку, що в ситуаціях коли продукт ширіться та росте перехід на МСА неминучий.

Netflix – хороший приклад реалізації філософії відповідальності за свій продукт. Процес коли діляться корисним кодом, а тим більше протестованим на бойових серверах, в виді бібліотек спонукає інших розробників розв'язувати схожі проблеми схожим чином, при цьому залишаючи можливість вибору іншого підходу при необхідності.

МСА любить контракти, але не любить оверхеди, тому використовує різноманітні шляхи управління цими контрактами. Такі шаблони як Tolerant Reader і Consumer-Driven Contracts використовуються в МС досить часто. Це дає змогу їм незалежно розвиватися. Перевірка Consumer-Driven контрактів як часть збірки програми збільшує впевненість розробників у тому, що сервери працюють вірно. Корисно коли це стає частиною збірки: сервіс збирається лише до того моменту, коли вимоги контракту виконані. Це елегантний спосіб обійти проблему YAGNI.

Напевне, апогей у децентралізованому управлінні – це метод, використаний Амазоном. Команди відповідають на всі аспекти стосовно програмного забезпечення, розробленого ними. Сюди входить навіть його цілодобова підтримка. Це не зовсім можна вважати нормою, адже така велика відповідальність може пагубно вплинути на розробників або на їх психічний стан при написанні коду, але все ж це призводить до того, що продукт стає більш якісним. Netflix також використовує даний метод. Адже уникнення підйому о 4 ранку – це чудовий стимул щоб приділяти більше уваги тому коду що пишеться.

Один із важливих моментів – така архітектура має на увазі, що всі ці МС повинні вміти працювати асинхронно. Але у сучасному світі потреба

писати асинхронний код чомусь викликає великі проблеми у багатьох розробників. Саме тому ми можемо спостерігати за успішним впровадженням МСА у проєкти компаній-гігантів, але не бачимо великих результатів у малих командах.

Більшість видатних компаній, а також стартапів розробляють програми опираючись на монолітну архітектуру. Такий початок зумовлений тим, що на початковому етапі набагато легше встановити моноліт і допомагати бізнесу розвиватися. Однак з виникненням перших проблем через розширення сервісу, через певний час, архітектура та код починають ставати дедалі складнішими. Тоді для підтримки проєкту необхідно все більше розробників. Одночасно з цим втрачається гнучкість та швидкість реагування на непередбачувані ситуації, що можуть виникнути в процесі роботи програми.

У архітектурі, основаній на МС окремі послуги підтримуються незалежними спеціалізованими командами, саме тому МСА сильно залежить від людей та процесів, залучених до проєкту.

РОЗДІЛ 3. МІКРОФРОНТЕНДИ

3.1. Чому код можна вважати монолітом

По своїй природі все frontend-програми – моноліти. Звичайно окрім тих які спеціально роблять на МСА. А це все тому що незалежні розробники пишуть код із використанням бібліотек, але їм необхідно використовувати одну версію цієї бібліотеки і тримати інших в курсі оновлень. Отже, пишучи таким чином код вони стають залежними один від одного. Можливо вони навіть використовують один репозиторій та одну версію збірки. Однак МС можуть спасти не лише бекенд, а і фронт.

Згадуючи, що таке МС – це техніка розробки програми таким чином, щоб розробники могли незалежно один від одного змінювати та деплоїти код, при чому це ніяк не відбивається на інших релізах.

Схема, що наглядно пояснює відмінність моноліту від МС

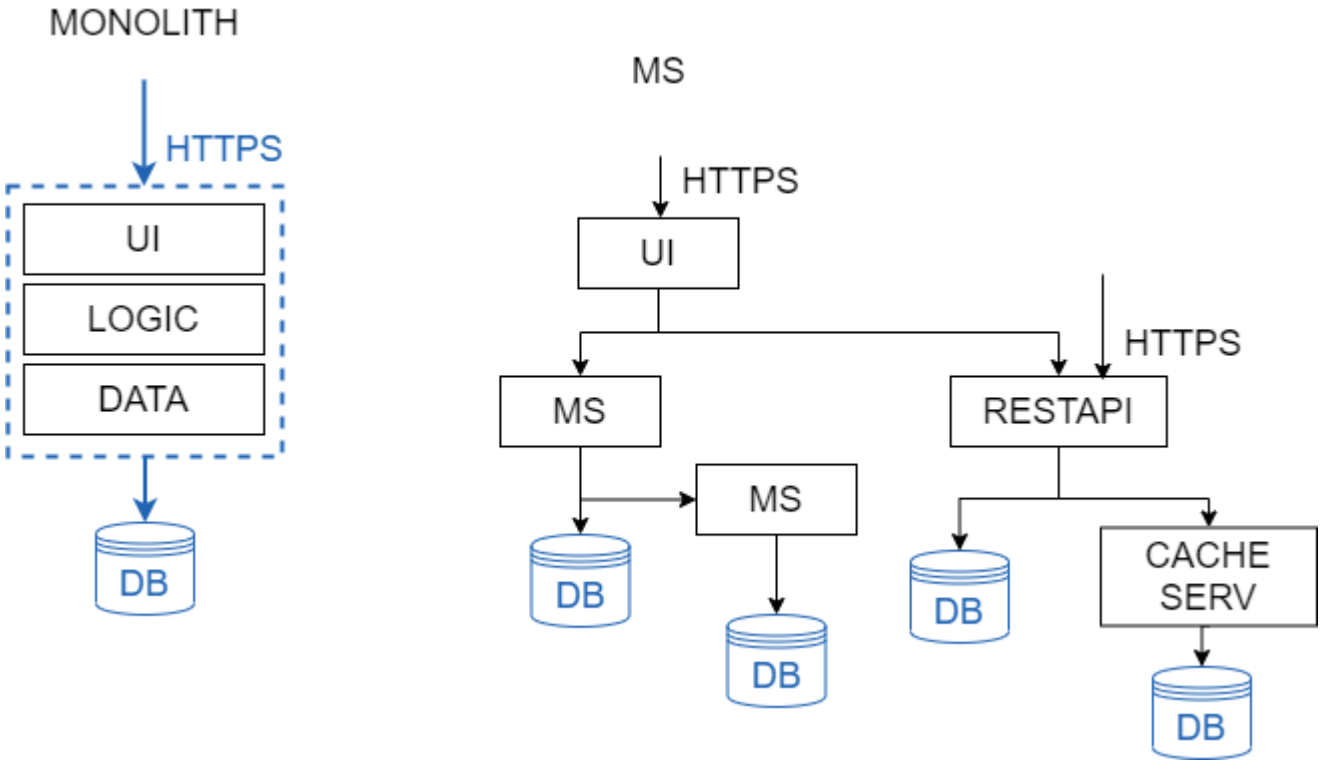


Рисунок 3.1. Різниця між МС та Монолітом

Бачимо, що кожен сервіс у системі МС є окремою програмою, окрім UI

– він залишився єдиним цілим. Якщо всі сервіси підтримуються однією командою, то є ризик що по мірі росту компанії, фронтенд розробники не зможуть поспівати за UI. У цьому і є головна проблема даної архітектури.

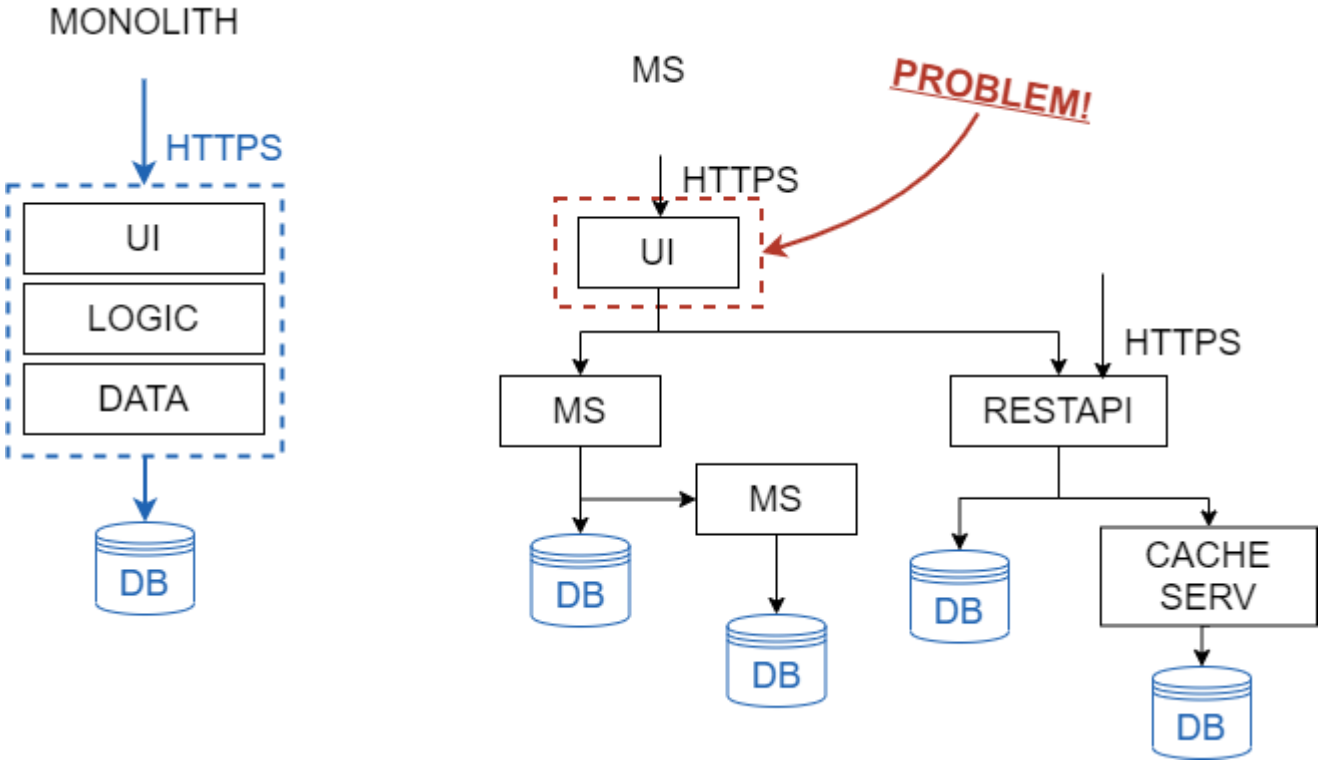


Рисунок 3.2. Виникнення несподіваної проблеми

3.2. Рішення: мікрофронтенд

Нехай спочатку є багато команд, що займаються розробкою МС, і кожен із них публікує свій API. І є окрема команда, яка займається розробкою кінцевого SPA, використовуючи API різних мікросервісів. При такому підході все працює, адже розробники МС знають все про їхню реалізацію, а розробники SPA знають всі тонкощі користувацької взаємодії.

Але з'являється проблема: тепер кожен фронтендер повинен знати всі нюанси роботи всіх МС. Кількість МС зростає, фронтендерів стає більше – і в результаті кількість людей у команді заважає продуктивності, і тако зникають взаємозаміна та універсальність.

Так ми підійшли до наступного етапу: модульної розробки. Нехай команда фронтенда поділилася на підкоманди. Кожна відповідає за свою

частину програми. Стало легше, але і цей підхід вичерпав себе через деякий час.

І ось чому:

- Усі модулі різнорідні, кожен зі своєю специфікою. Для кожного модуля краще підходять свої технології. При цьому вибір технологій – складна задача в умовах SPA.
- Оскільки програма SPA (тобто компілюється в єдину збірку), то одночасно можуть виконуватися лише розгортання всієї програми. Ризик кожного розгортання зростає.
- Управляти залежностями стає складніше. Різним модулям потрібні різні (і скоріш за все специфічні) версії взаємодій. Хтось не готовий перейти на оновлені API залежності, а хтось виправляє помилки у коді.
- Через пункт 2 реліз повинен бути одночасним, тому всі ждуть тих, хто відстає.

Описана вище проблема вирішується досить просто: завдяки мікрофронтендам. Аналогічне рішення вже давно та успішно застосовують при розробці бекенда. Тобто, потрібно взяти моноліт та розділити його на невеликі UI-елементи.

Однак, UI не зовсім схожий на сервіси, бо це інтерфейс між продуктом та кінцевим споживачем, отже він повинен бути продуманим та системним.

Принцип мікрофронтендів: розробка веб-сервісу як набір функцій, за які відповідають незалежні команди. У кожної із команд є своя задача та своє поле роботи, на якому вони спеціалізуються. Команда повинна бути кросс-функціональною та розробляти весь цикл програми: від БД до інтерфейсу користувача.

Уявимо, що ми розділюємо структуру монолітної програми по вертикалі, тобто по бізнес-функціям. Ми отримаємо декілька більш маленьких

програм з такою ж структурою, що й у монолітної програми. Але якщо ми додамо спеціальну програму поверх цих невеличких програмок, то користувачі будуть взаємодіяти з нею. Вона, в свою чергу, об'єднає UI тих маленьких програмок. Назвемо цей рівень зв'язуючим, адже він бере UI-елементи кожного МС та об'єднує їх в єдиний інтерфейс – ось найбільш пряма реалізація мікрофронтенда.

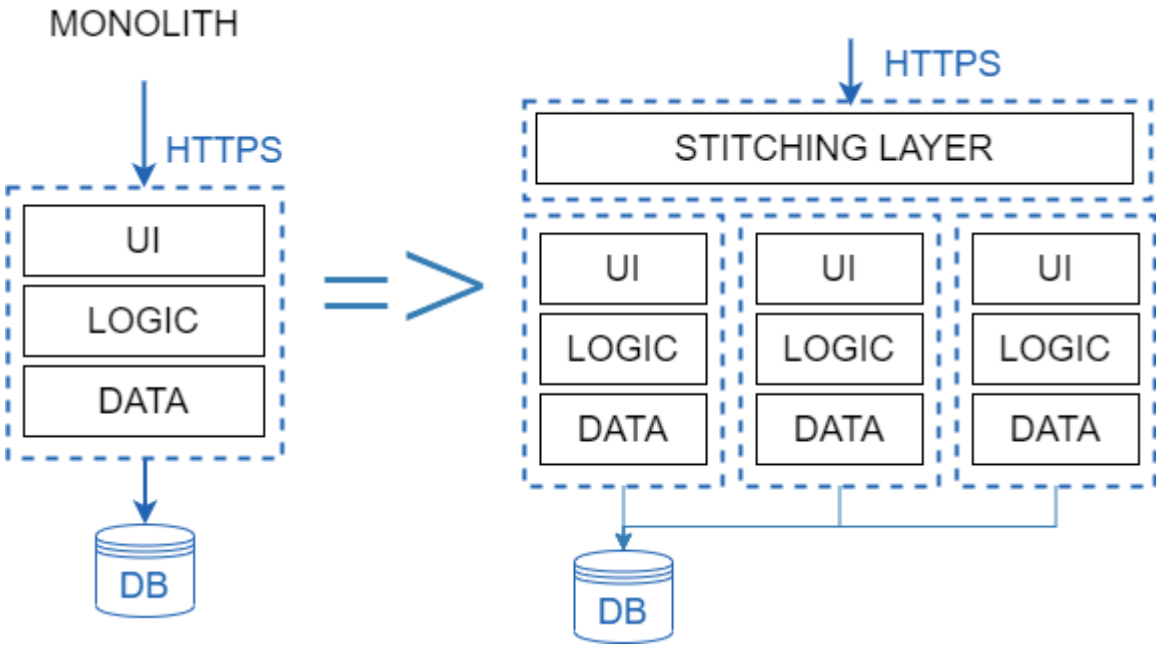


Рисунок 3.3. Розбиття моноліту на мікрофронтенди

Мікропрограма (далі МП) – утворені маленькі монолітні програмки. Це не просто МС, а автономні мікро програмки, адже кожна із них являє собою конкретну повноцінну бізнес-функцію.

3.3. Проблематика

Переводячи розробку програми у режим мікрофронтенду отримуємо наступні переваги:

- повністю ізольована частина UI, ніяк не залежна від інших. Радикальна ізольованість, буквально розробляється як окрема МП.
- кожен фронтендний МС відповідає за деякий набір бізнес-

функцій від початку до кінця

Тобто є повнофункціональним сам по собі.

- Може бути написаний на будь-яких технологіях.

У той самий час ми отримуємо певні труднощі. Їх зараз розглянемо детальніше. Багато людей починаючи розбивати монолітний фронтенд на мікросервіси стикалися з рядом проблем, іноді однією, іноді декількома, а іноді й усіма із наведених нижче.

3.3.1. Досягти цільної та узгодженої поведінки UI, коли у нас декілька абсолютно автономних МП.

Панацеї не існує, але ми можемо створити спільну UI-бібліотеку, яка також була б незалежним МП. У такому випадку, всі інші МП будуть залежні від цієї UI-бібліотеки, що повністю вбиває їх незалежність.

Інший варіант – створити спільні CSS-змінні на рівні кореня. Перевага такого рішення буде у тому, що ми отримаємо глобальну тему для всіх програм, яку можна налаштувати.

Або ж ми можемо зробити SASS-змінні та домішки загальними для всіх команд. Серед мінусів даного рішення є те що можуть попастися люди які не знають як працювати з такими змінними, та реалізація UI-елементів, яка постійно повторюється та потребує перевірки дизайну схожих елементів у всіх МП.

3.3.2. Переконатися, що одна команда не переписує CSS іншої команди

По-перше, можна задати межі областям видимості CSS за допомогою селекторів сформованих по назві МП. Помістивши це обмеження на зв’язуючий рівень, можна скоротити загальний час розробки, але водночас зросте відповідальність зв’язувального рівня.

По-друге, можна привести кожне МП до вигляду звичайного веб-

компоненту. Переваги такого підходу в тому, що обмеженням займатиметься браузер. Та, на жаль, налаштовувані елементи не підтримується браузерами на 100% - тим більше, якщо потрібна підтримка IE.

3.3.3. Зробити глобальну інформацію загальною для різних МП

Це одна із найбільш поширених проблем, але вирішується вона достатньо легко.

HTML5 має достатньо розширений функціонал, але не вивчений повністю усіма фронтенд-розробниками. Одна із таких функцій – кастомні події, які дають можливість робити інформацію спільною для МП.

У випадку якщо необхідна більш реактивна архітектура то можна реалізувати невеликий спільний Redux – так можна отримати більш тонкий обробник глобальних подій.

3.3.4. Якщо всі МП автономні, то як проводити маршрутизацію на стороні клієнта?

Розв'язання даної проблеми залежить від реалізації. У всіх основних сучасних фреймворках вже присутні механізми маршрутизації на стороні клієнта за допомогою стану історії браузера. Проблема у тому, яка програма відповідає за поточний route і коли саме.

Підхід спеціалістів полягає в тому, щоб створити загальний клієнтський маршрутизатор, відповідний лише за маршрути верхнього рівня, а інше віддати на відкуп відповідним програмам.

Наприклад, у нас є визначення маршруту /content/:id. Загальний маршрутизатор розв'яже проблему з /content, і отриманий маршрут буде переданий до ContentMicroApp, де ContentMicroApp – автономний сервер, який буде викликатись тільки з /:id.

3.3.5. А нам точно потрібна SSR (server-side rendering), та чи можлива вона при використанні мікрофронтендів?

Рендер на стороні сервера – справа непроста. Якщо ви хочете зв’язати МП за допомогою iframes, то необхідно забути про рендеринг на стороні сервера. Аналогічно, веб-компоненти для зв’язування не сильніші iframes. Однак, якщо кожна із МП здатна рендерити контент на стороні сервера, то зв’язуючий прошарок буде відповідати лише за об’єднання HTML-фрагментів на стороні сервера.

3.3.6. «Інтеграція з уже існуючим оточенням необхідна як повітря!» Так як же її провести?

Для інтеграції з існуючими системами доцільно використовувати «поступове введення»

Насамперед, потрібно реалізувати зв’язуючий прошарок так, щоб він мав функціонал прозорого проксі-сервера. Після цього ми зможемо позначити існуючу систему як МП (LegacyMicroApp), об’явивши до нього спеціальний роут. Весь трафік, що потрапляє на зв’язуючий рівень буде прозоро проксіюватися в системі, що існує, адже інших МП поки немає.

Наступний етап – поступове введення. Береться невеликий шматочок LegacyMicroApp, видаляється головна навігація та замінюється на залежність. Ця залежність – МП, реалізована за допомогою нової технології NavigationMicroApp.

Тепер LegacyMicroApp буде перехоплювати всі роути через залежність NavigationMicroApp, та обробляти їх вже всередині себе.

Потім аналогічно перероблюємо футер. Так ми будемо продовжувати відривати від LegacyMicroApp по шматочку, поки від нього нічого не залишиться.

3.3.7. Оркеструвати сторону клієнта так, щоб не доводилося кожен раз перезавантажувати сторінку

Зв’язуючий рівень розв'язує проблеми на стороні клієнта, але не на стороні сервера. На стороні клієнта, ми, завантаживши єдиний HTML, не можемо завантажувати при зміні URL окремі частини. Тобто, нам потрібен механізм, який завантажуватиме фрагменти асинхронно. Проблема у тому, що у цих фрагментів можуть бути залежності, які потрібно вміти вирішувати на стороні клієнта. Це означає, що мікро-фронтенд-рішення повинне мати механізм завантаження МП та введення залежностей (dependency injection).

Підбиваючи підсумки, маємо 2 області проблем:

- Сторона клієнта:
  - оркестрація
  - маршрутизація
  - ізоляція МП
  - взаємодія програм
  - єдність UI МП
- Сторона сервера:
  - серверний рендеринг
  - маршрутизація
  - управління залежностями

3.4. Гнучка та проста архітектура МП

3.4.1. Сторона Клієнта

Користуючись основними властивостями та вимогами було придумане рішення microfe (передловля фідбека).

Легше всього почати зі сторони клієнта, яка має 3 окремі основні структури: AppsManager, Loader, Router, а також однієї додаткової, MicroAppStore.

Нижче приведено рисунок, на якому вказано взаємодію між мікрофронтендами, яка виключає певні проблеми, зазначені вище.

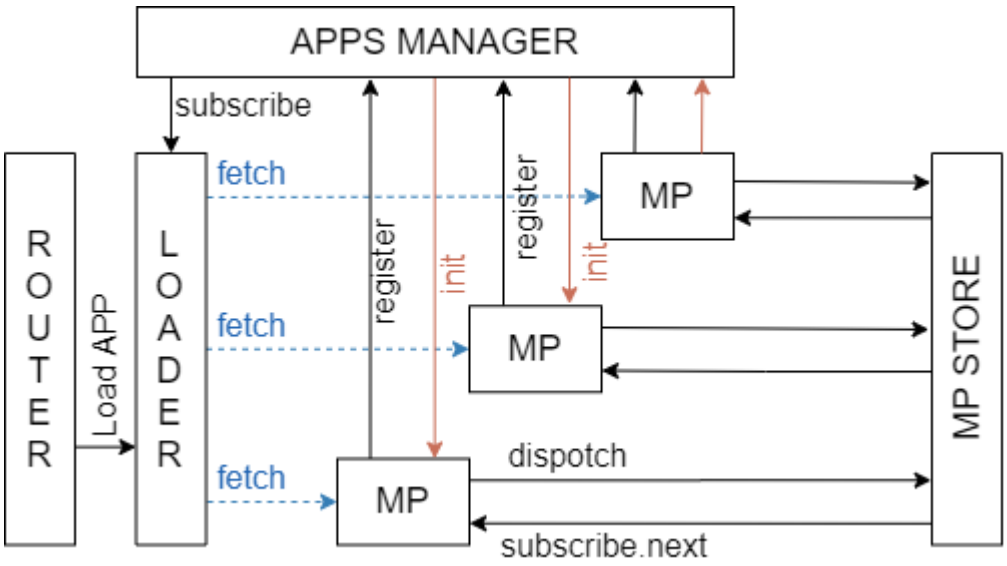


Рисунок 3.4. Схема сторони клієнта

- AppsManager – ядро оркестрації МП на стороні клієнта.

  Основна задача AppsManager – створення дерева залежностей. Як тільки всі залежності дозволені, AppsManager запускає МП.
- Loader – ще одна важлива частина оркестровки клієнтської сторони.

  Він відповідає за завантаження програм для клієнтської сторони.
- Router. Він введений для здійснення маршрутизації на клієнтській стороні.

  На відміну від звичайних маршрутизаторів сторони клієнта, цей має обмежений функціонал. Він оброблює не сторінки, а МП. Припустимо, у нас є URL /data/connect/47 і ContentMicroApp. У такому випадку маршрутизатор microfe обробить URL до /data/* та викличе частину ContentMicroApp /connect/47.
- MicroAppStore введено для рішення клієнтської взаємодії між МП.

  Він має схожий функціонал що і бібліотека Redux, але з одним

нюансом: він гнучкіший стосовно до асинхронних змін даних та оголошення reducer'а.

3.4.2. Сторона Сервера

Сторона сервера, можливо, трохи більш складна в реалізації, але має просту структуру. Вона складається із двох основних частин – StitchingServer та MicroAppServer.

1. MicroAppServer

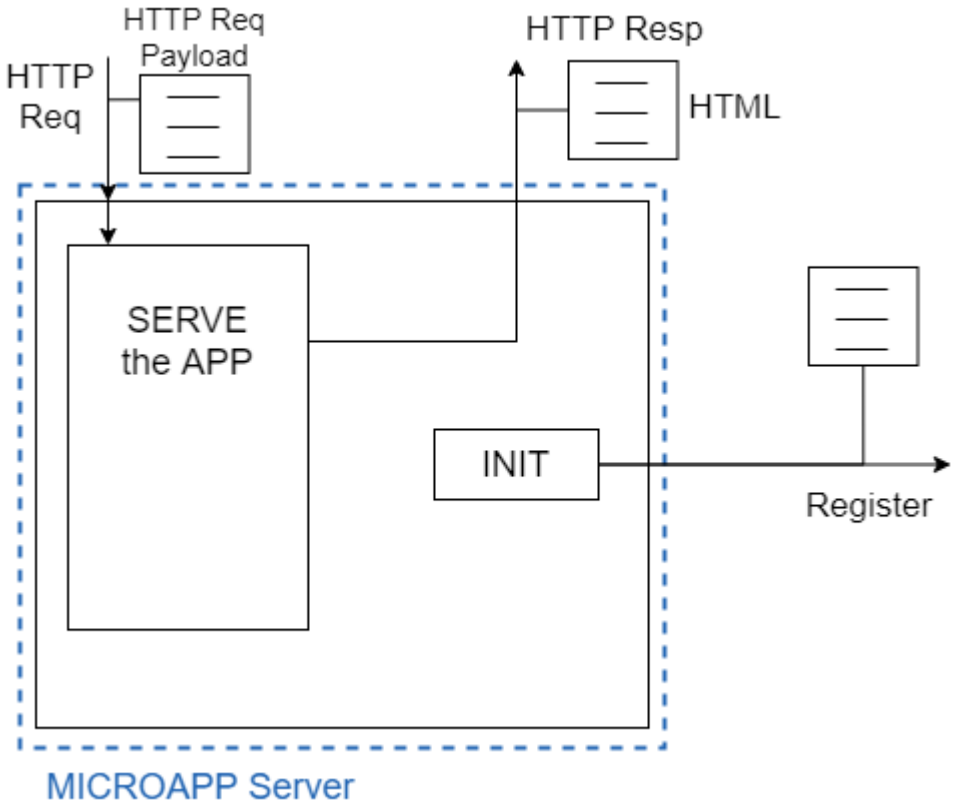


Рисунок 3.5. Схема сторони сервера, MicroAppServer

Мінімально можливий функціонал MicroAppServer можна виразити так: init та serve.

Коли MicroAppServer завантажується, перше що потрібно йому зробити – це викликати SticthingServer і зареєструвати ендпоінт з об'явленим МП. Воно визначає залежності, типи та URL схеми MicroAppServer.

2. StitchingServer

StitchingServer дає змогу зареєструвати endpoint в MicroAppServers.

Коли MicroAppServer реєструється в StichingServer, StichingServer записує оголошення в MicroAppServer.

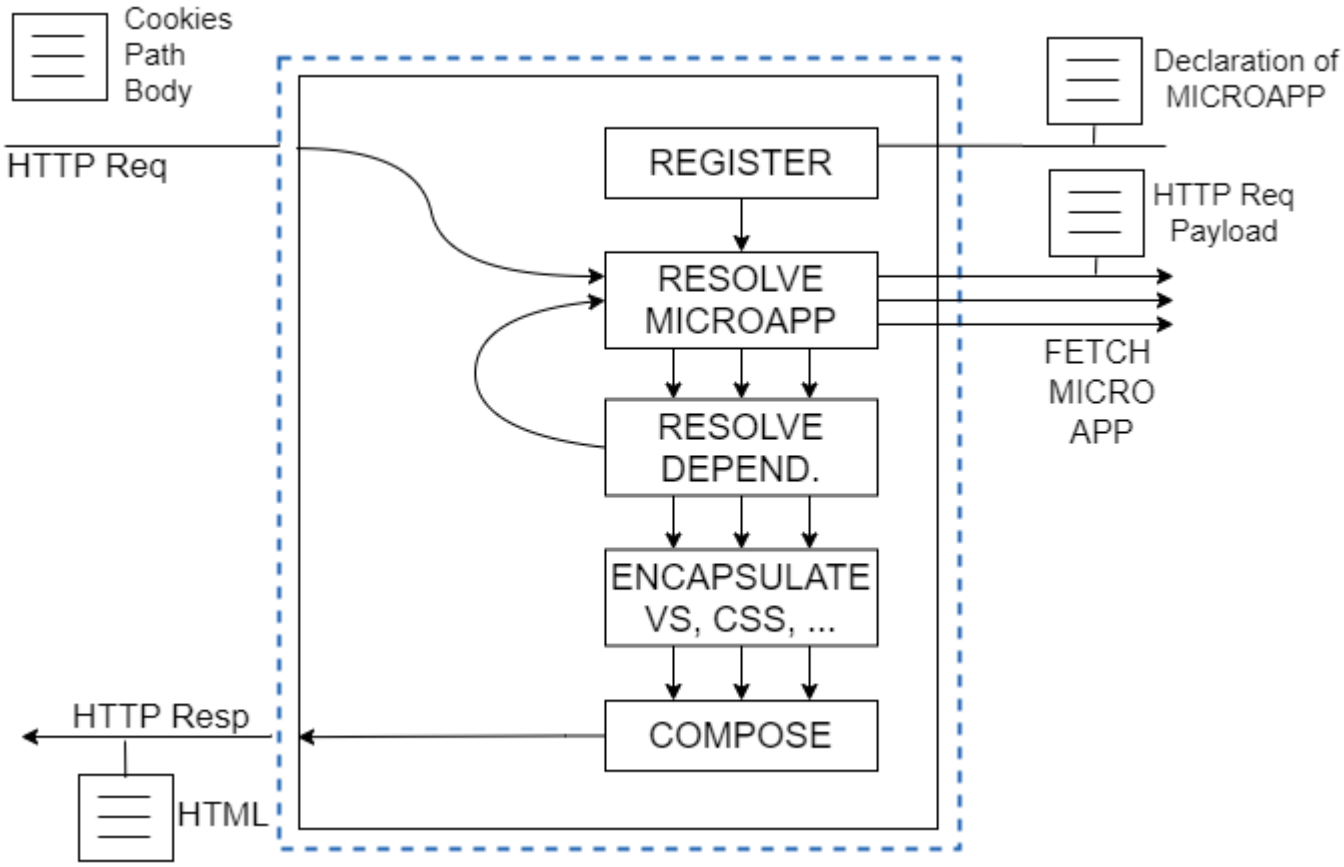


Рисунок 3.6. Схема сторони сервера, StitchingServer

Пізніше StitchingServer використовує оголошення для дозволу MicroAppServices від запиту URL.

Після дозволу MicroAppServer і всі його залежності, в назвах усіх відповідних шляхів у CSS, JS и HTML з'явиться відповідний публічний URL. Додатковий крок – Додавання до CSS-селекторів унікальний префікс MicroAppServer для уникнення конфлікту між МП на стороні клієнта.

Потім виконується головна задача StitchingServer: компоновка усіх отриманих частин і повернення цільної HTML-сторінки.

3.5. Переваги та недоліки мікрофронтенду

Як і в будь-якій системі, в МСА теж є свої недоліки та переваги. Тим більше коли справа стосується здавалося б неподільного – фронтенду.

Почнемо із переваг:

Найважливіше, що ми отримуємо від поділу моноліту на фрагменти – можливість вибору технології кожною командою окремо та прозоре керування залежностями.

Окрім цього також маємо:

- чітко розділені зони відповідальності
- незалежні видачі: кожен фрагмент може мати свій релізний цикл
- підвищення стабільності рішення в цілому, оскільки видача одних МП не впливає на інші
- можливість легко відкотити назад частини коду, або видавати їх частково на аудиторію.
- фрагмент легко розміщується у голові кожного розробника, що приводить до реальної взаємозаміни членів команди.
- крім того, кожен фронтендер може глибше зрозуміти всі тонкощі взаємодії із відповідним бекендом.

Рішення з мікрофронтендом виглядає чудово. Адже тепер кожен фрагмент може сам вирішувати коли деплоїтися, що і є ознакою справжнього МС. Крім того, фрагменти коду можуть самі обирати способи збірки, методи оптимізації тощо.

Мінуси фронтендних мікросервісів:

- Взаємодію між фрагментами неможливо забезпечити стандартними ламповими методами як DI.
- Як бути зі спільними залежностями? Адже розмір програми буде зростати дуже швидко, якщо їх не виносити із фрагментів.
- За роутинг в кінцевій програмі все одно повинен відповідати хтось один.
- Що робити, якщо один із фрагментів недоступний/не може з’явитися

ВИСНОВКИ ДО РОЗДІЛУ 3

Ще до того, як в 2016 році з’явилися МС, багато крупних компаній намагалися розв'язувати схожі проблеми. Наприклад, Facebook з його BigPipe.

Зараз ідея набирає обороти. Компанії найрізноманітнішого масштабу цікавляться цією темою, інвестуючи в неї гроші та час. Наприклад, Zalando надала відкритий код свого рішення Project Mosaic. Описаний тут метод і їх виконуються з аналогічними підходами, але є кардинальна відмінність. Якщо microfe використовує повністю децентралізовану маршрутизацію для більшої незалежності кожної МП, то Project Mosaic надає перевагу централізованій маршрутизації та визначенню шаблону для кожного маршруту.

Існують і інші підходи, а саме, використання ifram’ів в якості зв’язуючого шару – очевидно, не на стороні серверу, а на клієнтській. Це дуже просте рішення, яке не потребує особливої серверної структури. Воно може бути реалізоване фронтенд-командою самостійно, отже створює менше організаційних проблем для компанії і коштує дешевше.

Також є фреймворк single-spa. Проєкт розрахований на згоді про назву кожної програми для дозволу і завантаження МП. Легко зрозуміти ідею та наслідувати шаблони. Тому фреймворк може бути корисним для ознайомлення та експериментів над системою у вільний час в локальному середовищі. Великий мінус у тому, що доведеться будувати кожне МП окремим шляхом, а інакше фреймворк може його не прийняти.

У МСА неможливо уникнути появи центральних сервісів, дані із яких необхідні усім іншим. Наприклад, сервіс локалізації, який зберігає переклади. Якщо кожен МС окремо почне лазити за цими даними на сервер, то ми отримаємо просто вал запитів при ініціалізації.

Тож для розв'язування цієї проблеми було розроблено реалізацію NativeJS сервісів, які надають доступ до таких даних. Це дало можливість не створювати зайвих запитів, і кешувати дані. У деяких випадках навіть

виводити такі дані раніше на сторінку в HTML, щоб взагалі уникнути таких запитів.

Думаю, що з часом тема мікрофронтендів буде розглянута програмістами більш детально. Якщо на неї почнуть звертати увагу все більше і більше компаній, то така концепція стане методом розробки в великих командах за замовчуванням. Для кожного фронтенд-розробника буде корисно уже в найближчий час ознайомитися з даною архітектурою.

Я не стверджую, що стиль МС це інновація. Його корені ідуть далеко в минуле, як мінімум до принципів проектування, які використовує Unix. Але вважається, що досі недостатньо людей беруть до уваги даний стиль архітектури. Якщо люди зможуть дізнатися більше про переваги даного стилю, то більшість програм отримають переваги.

РОЗДІЛ 4. РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

4.1. Опис інструментів розробки

В рамках даної дипломної роботи планується реалізувати веб-сервіс з мікросерверною архітектурою для нейромереж загального призначення, клієнтська частина. Тобто, фронтенд.

У попередньому розділі наведено перелік недоліків та переваг мікросервісного фронтенду, з чого робимо висновок що в даній роботі використовувати МС-фронтенд недоцільно. Припустимо що проєкт розробляється для невеликої компанії, або для локального використання. Тому тут буде створено звичайний монолітний, але модульний проєкт.

У випадку, якщо відбудеться аналогічно із ситуацією Убера, та програма розростеться, то доведеться переписати на МС звичайно, але поки що працюватимемо в рамках моноліту.

Для створення фронтенду обрано мови HTML для розмітки веб-сторінки, та створення елементів, з якими буде взаємодіяти користувач. Це не мова програмування, але саме вона дозволяє розбивати сайт на області та додавати в них елементи. Цією мовою виконана значна більшість сторінок в інтернеті. Разом с HTML завжди згадуються CSS. Це теж не мова програмування, а лише каскадна таблиця стилів.

Саме CSS спрощує роботу по розробці веб-сторінок, так що не треба писати щоразу одне і те ж – достатньо просто об’єднати елементи в один клас, і їм присвоїться окремий параметр, наприклад, колір або прозорість. У своїй роботі я часто використовую положення, відступи та саме прозорість для розробки елементів.

Отже, HTML та CSS контролюють те, що ви бачите на веб-сторінці. Перший відповідає за вміст, другий за оформлення. Тут зрозуміло.

Зараз, з розвитком технологій, дизайну та потреб людей вже не

достатньо звичайних вищевказаних технологій. Користувач завжди хоче, щоб було щось цікаве. Щоб сонце світилося, карусель крутилася, а сторінка змінювалася з мірою взаємодії користувача з нею. Для таких випадків і було створено на цей раз вже мову програмування Javascript. Це саме все відповідає за те, щоб меню зверху випадало коли сторінка промотується вниз, щоб анімації крутилися, а користувацькі дані не пропадали безслідно. Варто відзначити що дана мова програмування не має нічого спільного зі співзвучною їх – Java. Єдине, що можна назвати у них спільного, це назва.

Також було розглянуто варіант використання готових шаблонних сайтів, або платформ для спрощеного сайту, але жодна не задовольняла вимогам безпечності та підтримки мікросервісного бекенду, який продемонструє мій колега по темі диплому. Тож вибір пав саме на комбінацію трьох мов: HTML, CSS та JavaScript.

4.2. Аналіз оформлення популярних та корисних сайтів

4.2.1. Netflix

Оскільки при згадуванні МСА всі хто в цьому розуміється відразу згадують Netflix, то почнемо символічно з перегляду його дизайну.

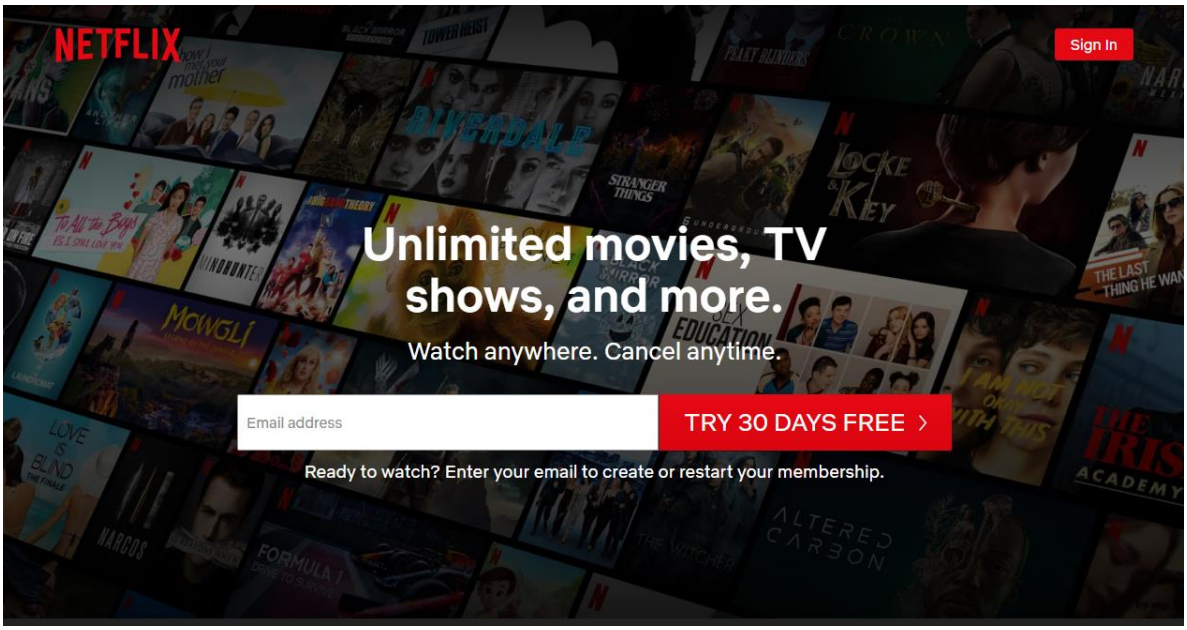


Рисунок 4.1. Домашня сторінка Netflix [11]

В шапці видно логотип, при натисканні на який переходить на головну сторінку, тобто цю ж. на самій же сторінці коротко і по ділу все. Можна відразу починати співпрацювати з ними, бачачи пропозицію спробувати їх сервіс безкоштовно впродовж місяця. Також присутнє пояснення, чим компанія займається та що вона може запропонувати користувачу.

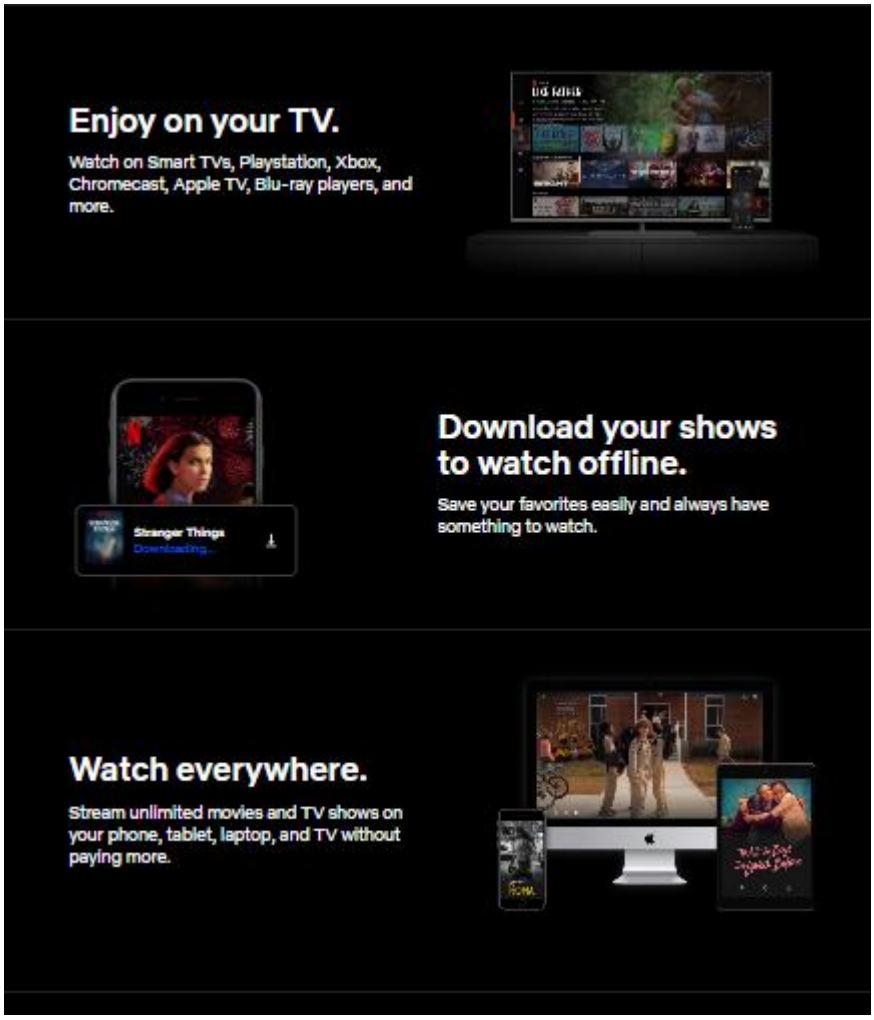


Рисунок 4.2. Домашня сторінка Netflix, основна інформація [11]

Прокрутивши нижче можна побачити основну інформацію, переваги. Не переходячи на інші сторінки маємо доступ до інформації – це зручно.

Проте, на мою думку, тут забагато пропусків та повітря. Доводиться багато промотувати щоб прочитати корисну інформацію.

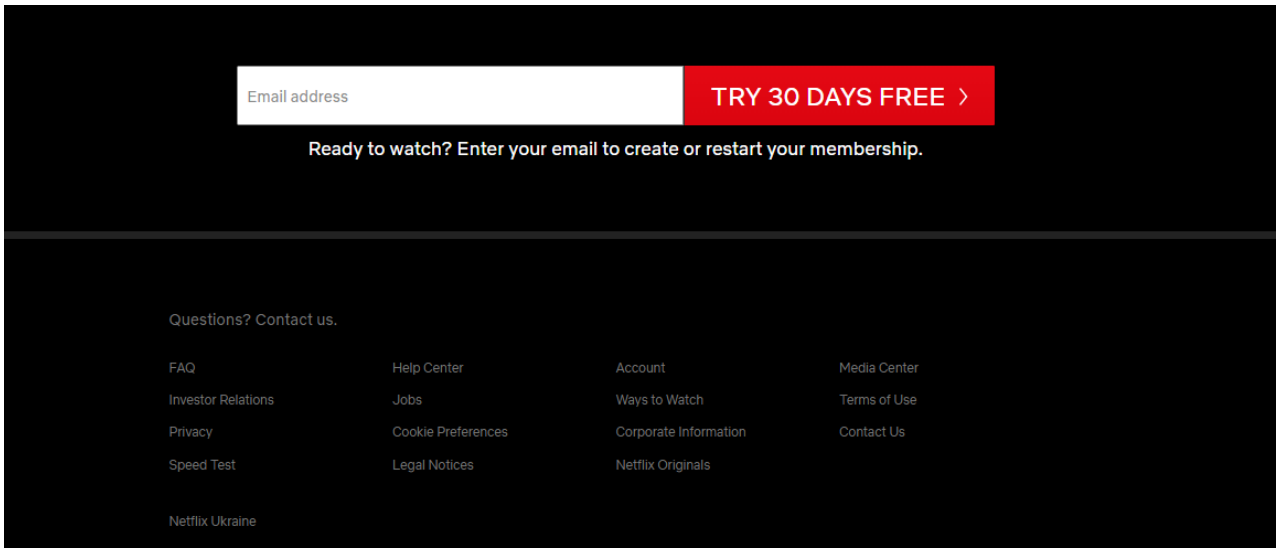


Рисунок 4.3. Домашня сторінка Netflix, кінець [11]

Знизу бачимо ще раз заклик приєднатися до співпраці, що є хорошим рішенням, адже користувач вже ознайомлений із основною інформацією. Також в кінці знаходиться копірайт.

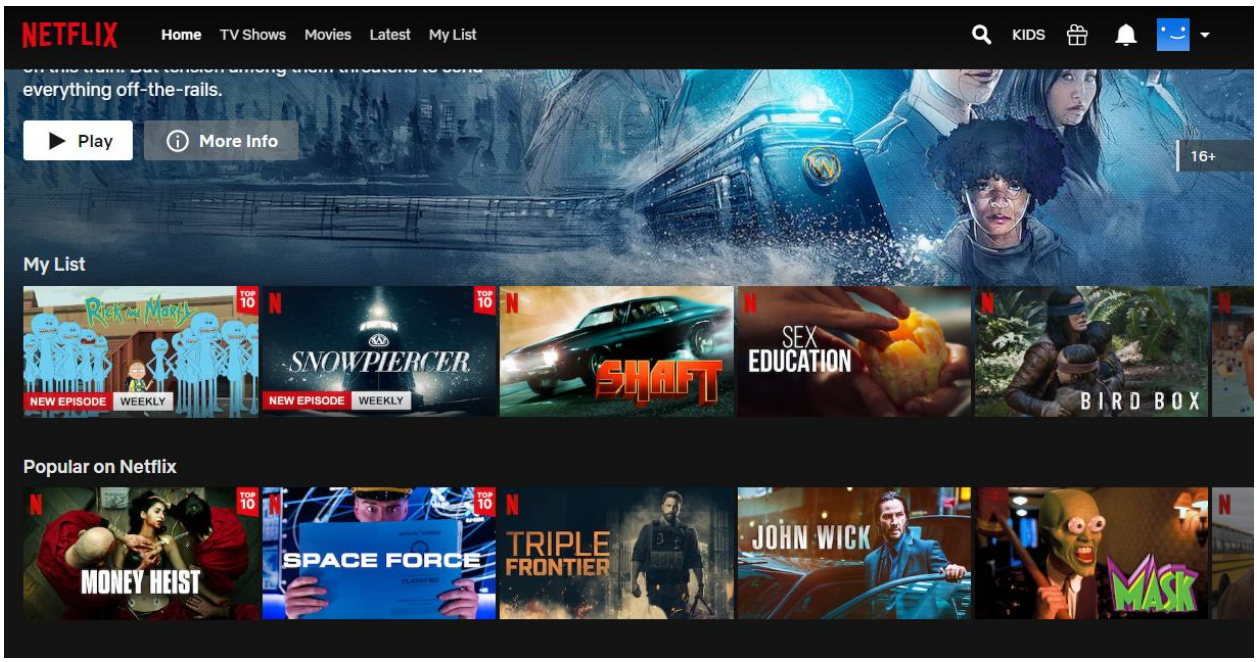


Рисунок 4.4. Головна сторінка для зареєстрованих користувачів [11]

Запитавши про переваги у кіно/серіалах при реєстрації та першому вході в акаунт, тепер на головній сторінці показуються матеріали, з урахуванням введених користувацьких переваг.

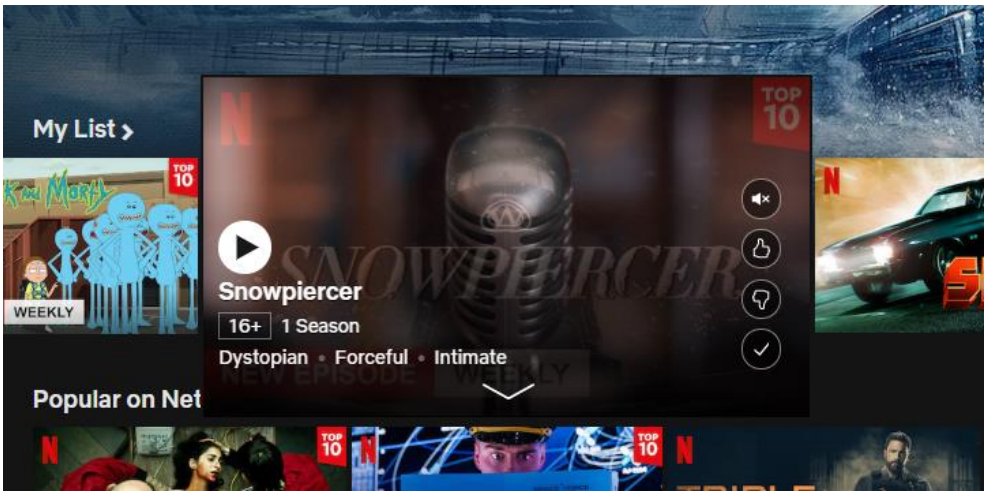


Рисунок 4.5. Головна сторінка, анімація [11]

При наведенні курсора бачимо гарну анімацію, при чому відео починає програватися, при цьому робота сайту не зупиняється, тобто навіть якби з відео була проблема все було б добре, тому що це не вплинуло б на сайт.

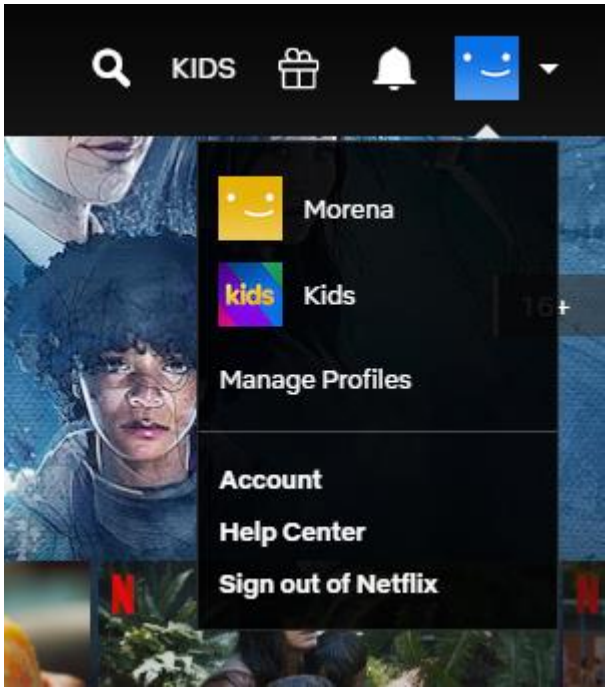


Рисунок 4.6. Налаштування акаунта [11]

Зручне випадаюче меню з достатнім переліком функцій.

Також бачимо на сайті зручний акаунт та його налаштування (рисунок 4.7(1)). Вся інформація перелічена, персональні дані сховані за зірочками або крапочками, присутня можливість їх змінити не відходячи далеко. Також

присутні налаштування окремих профілів(рисунок 4.7(2)).

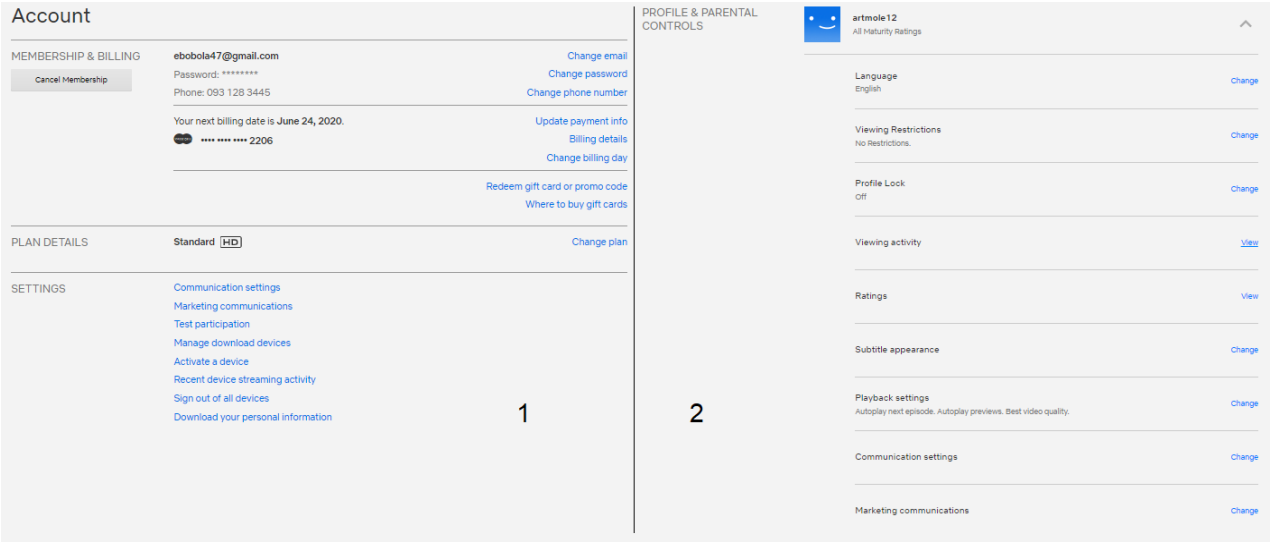


Рисунок 4.7. Налаштування аккаунту [11]

Отже, у Netflix дуже приємний та зручний сайт, яким можна з радістю користуватися та радити друзям. Оскільки сайт більше спрямований на розважальний характер, то й оформлення виконане у відповідному стилі.

4.2.2. PayPal та Google Fonts

Переглядаючи ці 2 сайти звернулася увага більше на меню, ніж на загальне оформлення, тому було вирішено об’єднати їх в один підпункт.

Тож, розглянемо те, що особливо сподобалося.

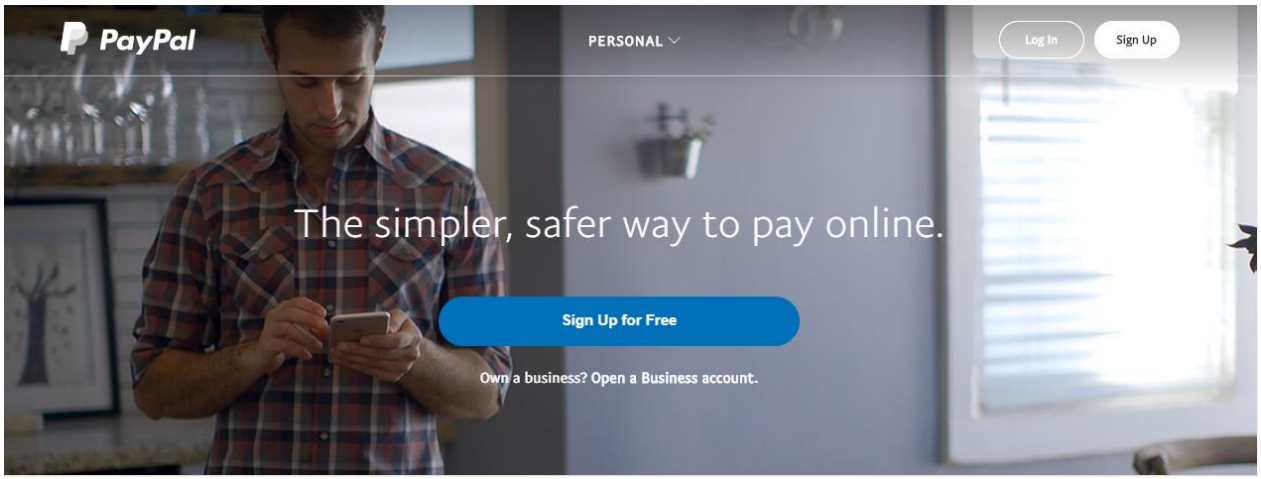


Рисунок 4.8. Шапка PayPal [10]

Особисто мені сподобалося темне фото на фоні шапки. Воно задає атмосферу робочого середовища. Це ідеально співпрацює із метою створення

даного сайту. Також привернуло увагу верхнє меню, яке напівпрозоре. Справа у кутку гарні заокруглені кнопки – зараз подібне вважається стильним і багато веб-сторінок та програм використовують саме такий дизайн кнопок.

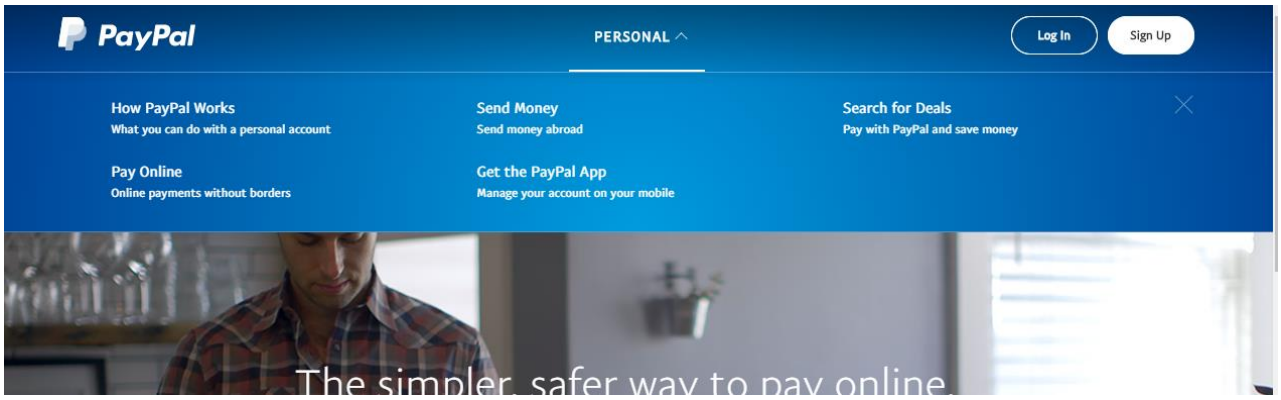


Рисунок 4.9. Зручне меню Personal [10]

При наведенні на напис посередині бачимо хороше анімоване рішення показати більше щодо компанії та її можливостей.

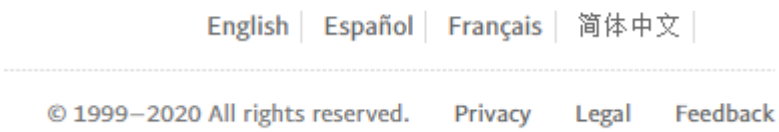


Рисунок 4.10. Вибір мови [10]

Знизу – меню зміни мови. Корисно для міжнародних організацій, показує клієнтоорієнтованість.

Тепер декілька слів про Google Fonts.

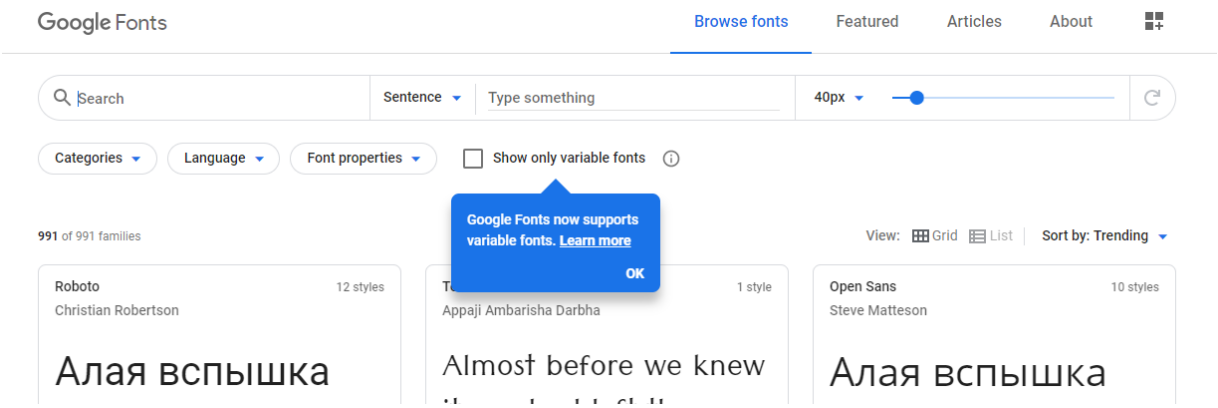


Рисунок 4.11. Загальний вигляд сайту Google Fonts [12]

Можна побачити, що функціонал сайту достатньо великий, що і видно з верхньої шапки з меню. Спочатку основна інформація, така як логотип, та основні функції, кнопка «про нас». Нижче розташовано поле для пошуку шрифтів – основної задачі даного сайту.

Найбільшою перевагою є те, що сайт зроблений без зайвих функцій, з мінімальним наповненням, при цьому максимально реалізовуючи поставлену перед ним задачу.

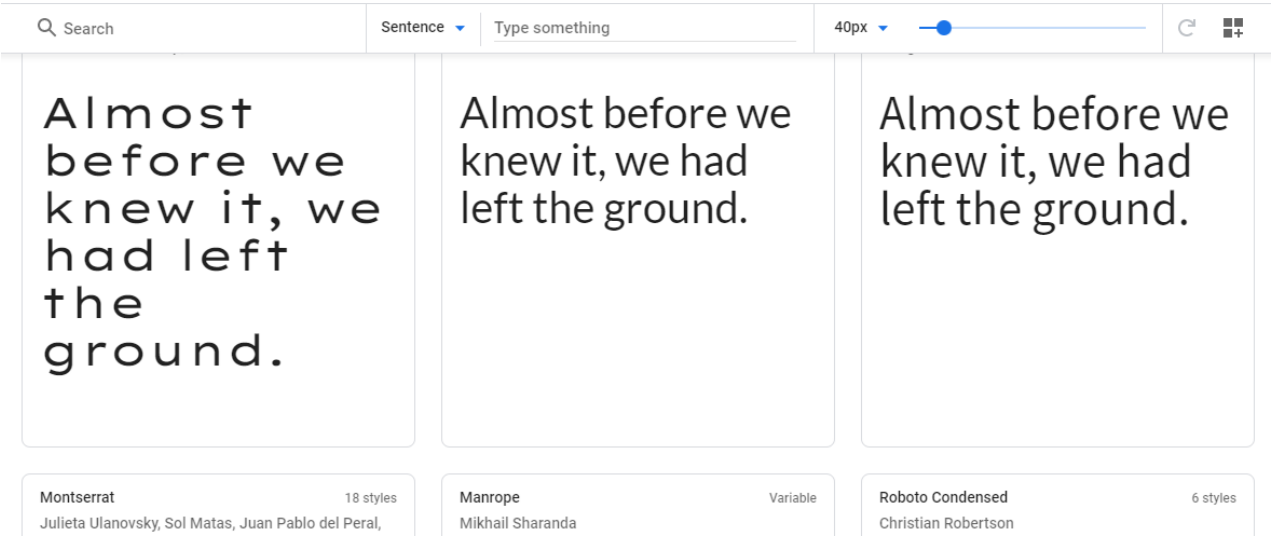


Рисунок 4.12. Стрічка налаштувань закріпилася зверху [12]

Відмотуючи вниз, бачимо що область налаштувань пошуку займає місце меню, та закріплюється зверху. Таким чином можна у будь-який момент змінити налаштування пошуку.

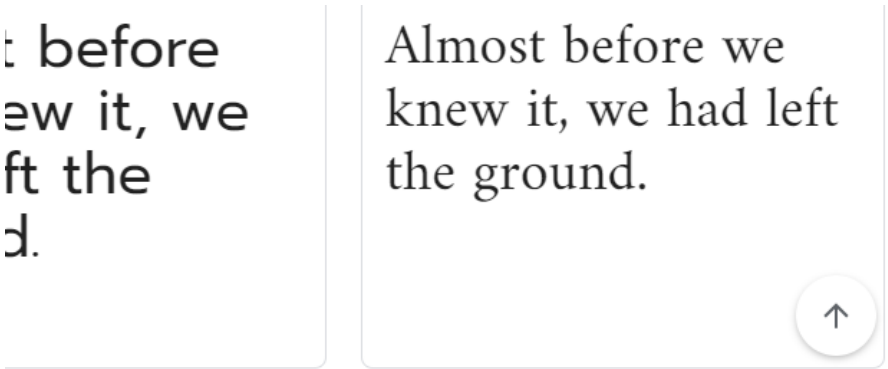


Рисунок 4.13. Кнопка «повернутися догори» [12]

Кнопка «Повернутися наверх» дуже корисна на подібних сайтах, які побудовані на принципі послідовного відображення даних, та на яких може

виникнути потреба повернутися до функціонала, що залишився вгорі.

Говорячи про авторизацію не можна оминути таку річ, як реєстрація або вхід на сайт за допомогою популярних соціальних мереж, таких як Facebook, Twitter, і найчастіший спосіб – через гугл-аккаунт. Особисто у мене на більшості сайтів здійснено вхід саме через аккаунт Google. Це зручно, адже не потрібно тримати в голові логін і пароль кожного разу, коли я хочу десь зареєструватися. До того ж при вході через аккаунт Google у більшості випадків автоматично прив'язується пошта, на яку потім можна отримувати корисні листи. Але, нажаль, частіше всього вони не корисні, а просто спам із нагадуванням про сайт.

Тож, переглянувши популярні сайти, та виділивши особливості цих трьох, можна починати проектувати дизайн свого сайту.

4.3. Дизайн власної сторінки.

4.3.1. Кольорова гамма сайту

Перед тим, як створювати сайт не так важливо продумати вміст та зміст сторінки, як кольори, які потім будуть всюди використовуватися.

Тому перше, чим я зайнялася – це пошук кольорової гамми, за допомогою сайту color.romanuke.com. Вибір зупинився на яскравій весняній палітрі, вказаній на рисунку 4.15.

До неї було підібрано монохромні кольори, та створено градієнт переходу від кольору фона до кольору шапки.

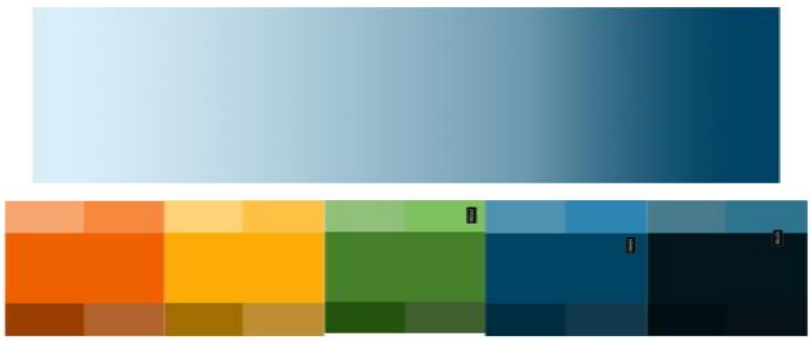


Рисунок 4.14. Набір кольорів для сайту

Рисунок 4.15. Запозичена палітра кольорів [13]

Після підібраного стилю можна починати розробляти дизайн сайту і саму веб-сторінку, не оминаючи увагою такі речі як гармонія елементів тих чи інших кольорів.

4.3.2. Розробка каркасу дизайну сайту

Перше, що бачить користувач, переходячи по посиланню – сторінка привітання. У Netflix вона відрізняється від головної сторінки зареєстрованого користувача. І основна відмінність – наявність можливості зареєструватися чи увійти в акаунт. Тобто, необхідно створити першу сторінку з шапкою, та кнопками стосовно входу в систему.

Оскільки мені сподобався дизайн сайту PayPal на цьому етапі, то буду робити щось схоже. Тому необхідно визначитися із фоновим зображенням, логотипом та розміром шапки.

За тестовий пристрій було взято планшет із середнім розміром екрана,

щоб дизайн можна було легко адаптувати як до комп'ютерної версії з більшим екраном, так і до мобільної, з меншим.

Для пошуку зображення використано сайти з creative common правами, однак поки нічого знайдено не було, тому вирішено поки що створити каркас сайту без кольорів та фонових зображень.

Рисунок 4.16. Макет стартової сторінки

Видно аналогію із приведеними вище сайтами. Також було обрано кругловаті кнопки, тому що це зараз вважається стильним, і використовується багатьма програмами і веб-сервісами.

У полі «назва» буде назва веб-сервісу, а під нею короткий опис що він із себе являє, і навіщо його можна використовувати. Потім користувачу надається заклик до приєднання.

Пунктирною лінією показано уявні межі областей, як текстових, так і шапки. На ній зараз і зупинимося детальніше.

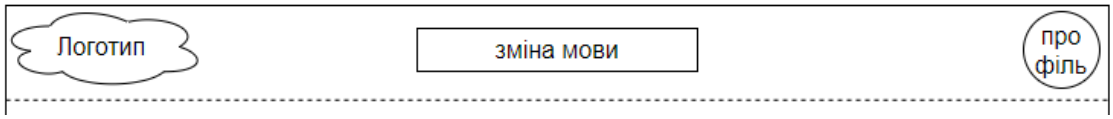


Рисунок 4.17. Макет шапки

Шапка статична, і залишається зверху впродовж того, як сторінка спускатиметься вниз. Схожа ідея використана на сайті Google Fonts.

Тож, щоб шапка була досить функціональною, на неї розміщено наступне:

- логотип – щоб користувачі не забували де вони знаходяться. При натисканні на нього відбудеться перенаправлення на головну сторінку
- Функціонал для зміни мови – щоб користувач міг одразу працювати у зручному йому оточенню.
- Профіль – для зареєстрованих користувачів там доступ у кабінет, а для тих, хто тільки що потрапив на сайт – поле для реєстрації або входу в систему.

Рисунок 4.18. Функціонал вибору мови

Якщо користувач вже увійшов до системи, то побачить такий профіль як на наступному рисунку (рисунок 4.19).

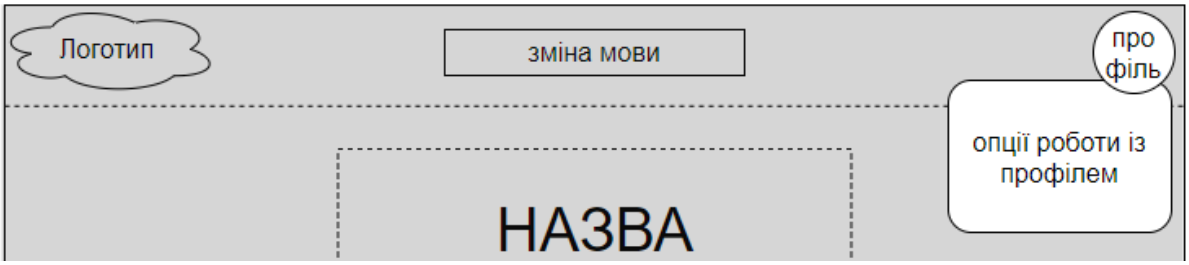


Рисунок 4.19. Профіль для авторизованого користувача

Однак, коли він не зареєстрований то система перенаправить користувача на сайт реєстрації, який виглядатиме наступним чином.

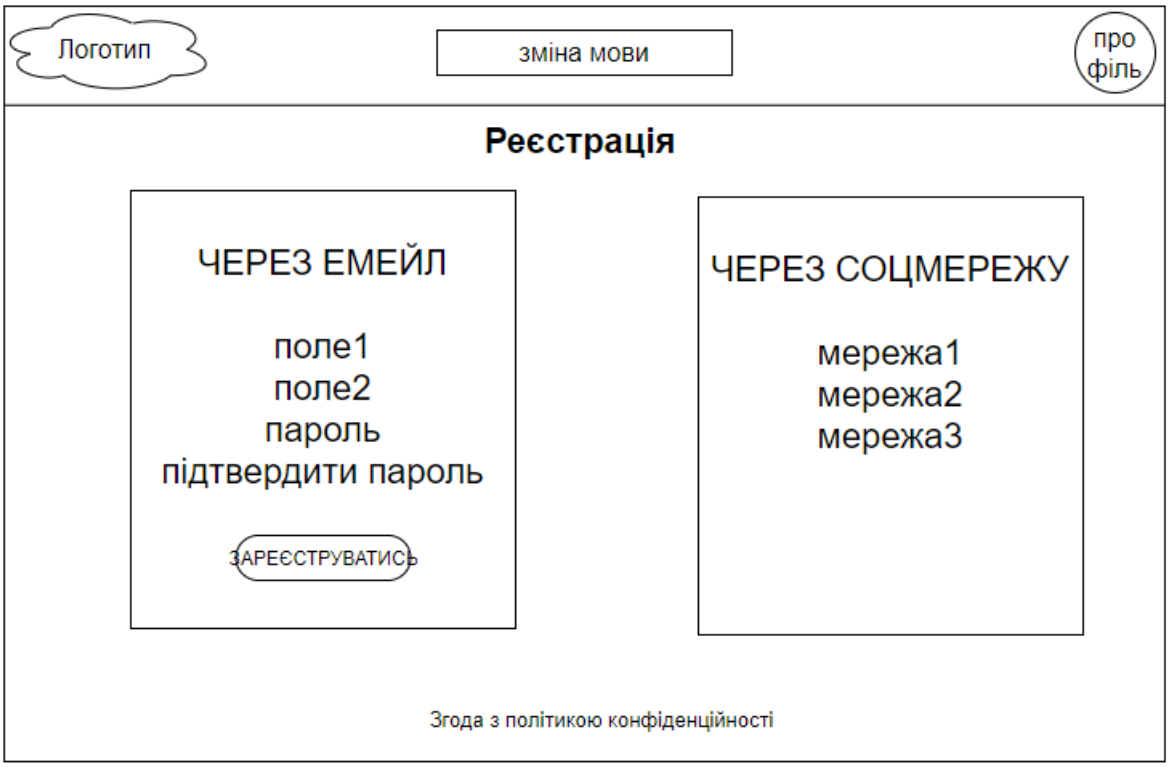


Рисунок 4.20. Макет поля реєстрації

Бачимо, що у користувача має бути можливість як зареєструватися по-швидкому, тобто через соціальні мережі, так і звичайним способом, використовуючи пошту.

Якщо користувачу цікаво дізнатися більше про сервіс, на веб-сторінку якого він потрапив – то він може відмотати сторінку вниз, та почитати основну інформацію.

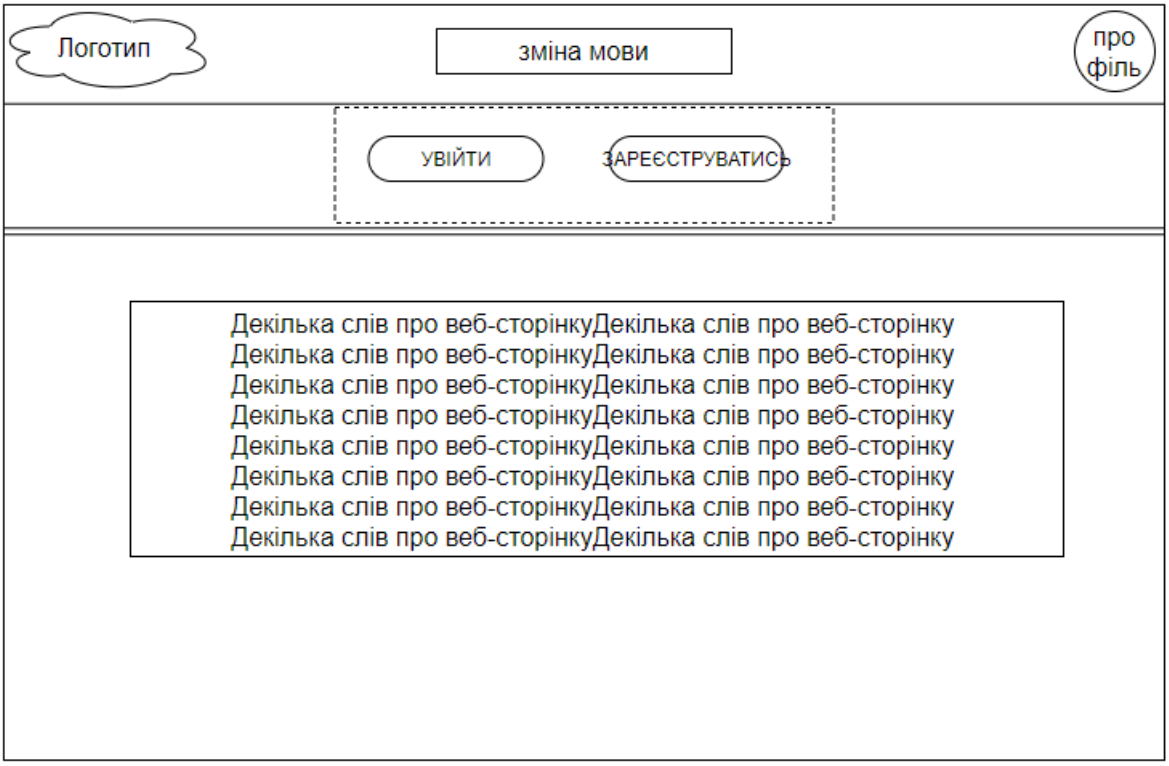


Рисунок 4.21. Прокручена вниз стартова сторінка

Після того, як у користувача склалося конкретне уявлення про те, хто ми, чим займаємося, і що він може у нас робити, йому ще раз пропонують увійти в систему або зареєструватися.

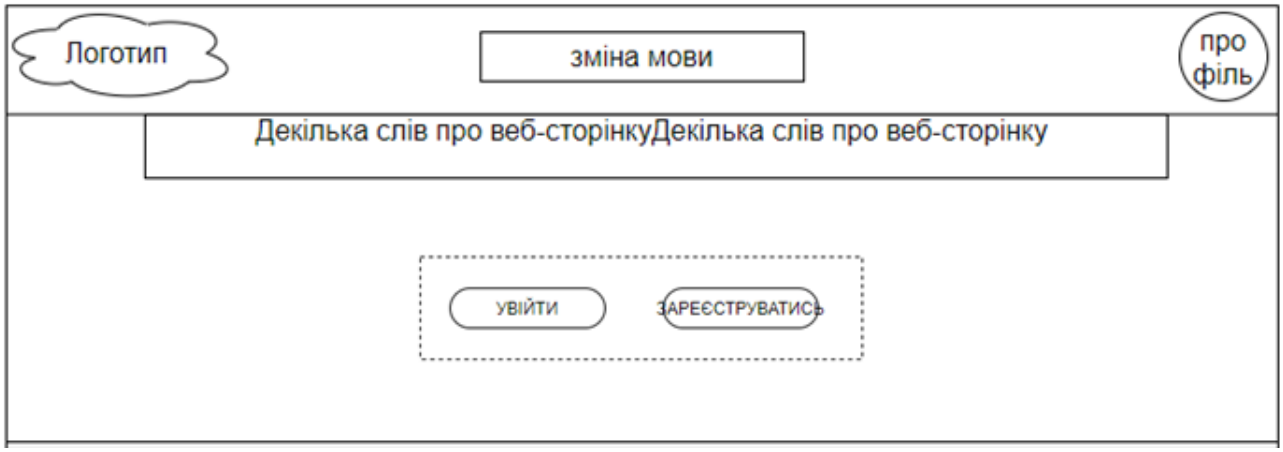


Рисунок 4.22. Повторний заклик до реєстрації

А у самому низу сторінки питання співпраці, посилання на соцмережі та відкритий ресурс коді, і копірайт.

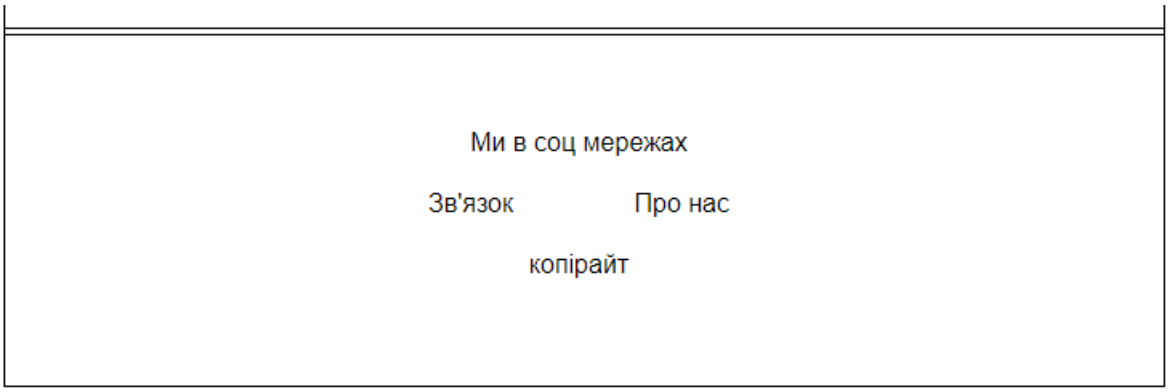


Рисунок 4.23. Поле копірайту

Ще одна із непомітних, але важливих речей при розробці сайту – це обробка помилок. Адже набагато приємніше, коли при невдалій обробці запиту вискакує не автоматичне повідомлення браузера, а розроблена з душею сторінка з помилкою, можливо навіть підказками по їх вирішенню, або засобом як повідомити про цю помилку.

Тому тут поки все просто, враховуючи те, що в макеті не використовуються ні кольори, ні малюнки, тому каркас інтуїтивно зрозумілий.

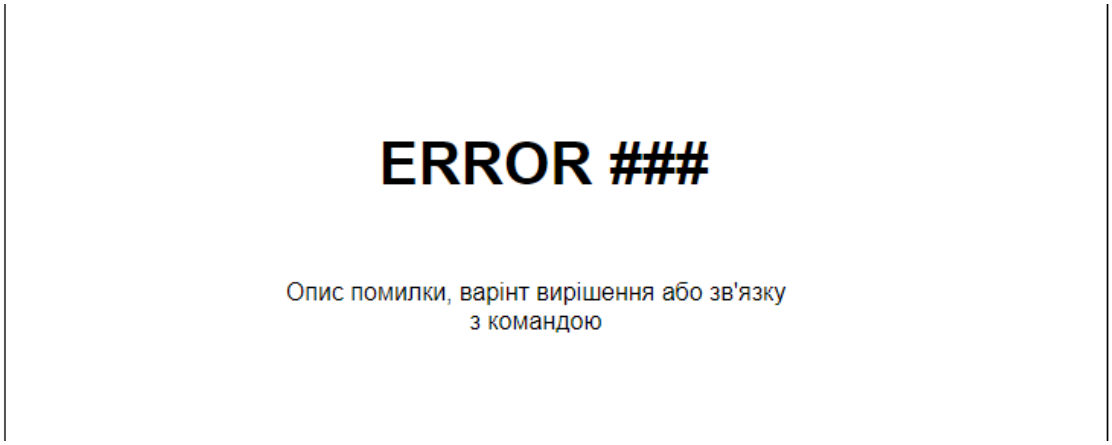


Рисунок 4.24. Макет повідомлень про помилку

Також важливою частиною веб-сервісу є представлення функціоналу, який там надається. Тому матимемо наступні сторінки для роботи із нейромережами широкого призначення:

1. Сторінка «Проєкти»

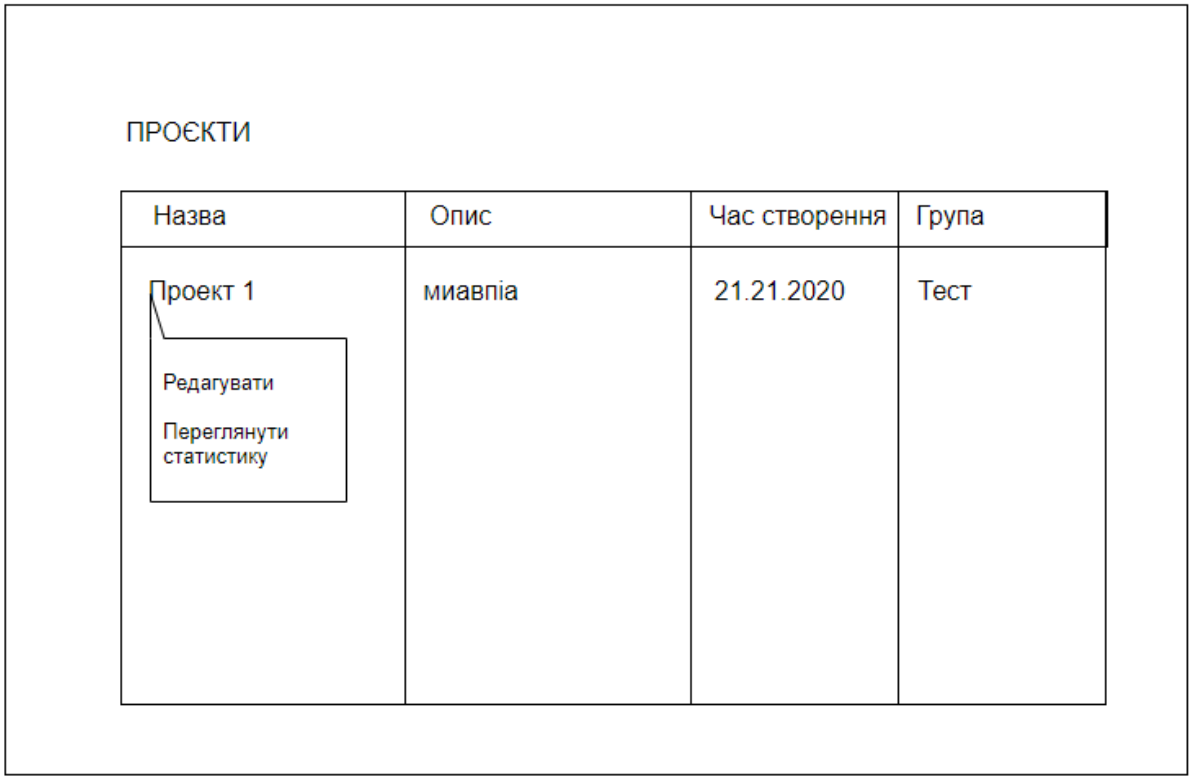


Рисунок 4.25. Проєкти користувача

На даній сторінці здійснено перелік всіх створених раніше проєктів. Цей функціонал має вигляд таблиці, з наступними полями:

- Назва проєкту – поле, що містить назву проекту, введену користувачем.
- Опис проєкту – короткі або не дуже підсумки щодо проєкту. Для чого робиться, що з ним робити далі. Поле для будь-яких текстових коментарів до даного проєкту.
- Час створення – дата і час створення проєкту. Зроблено для зручності систематизації проєктів користувача.
- Група – назва предметної області, до якої відноситься даний проєкт. В одній групі може бути декілька проєктів, але в одного проєкта лише одна група. Створено для зручності спостерігання за навчанням нейромережі, та використовується при побудові графіку.

2. Сторінка «Створення/Редагування»

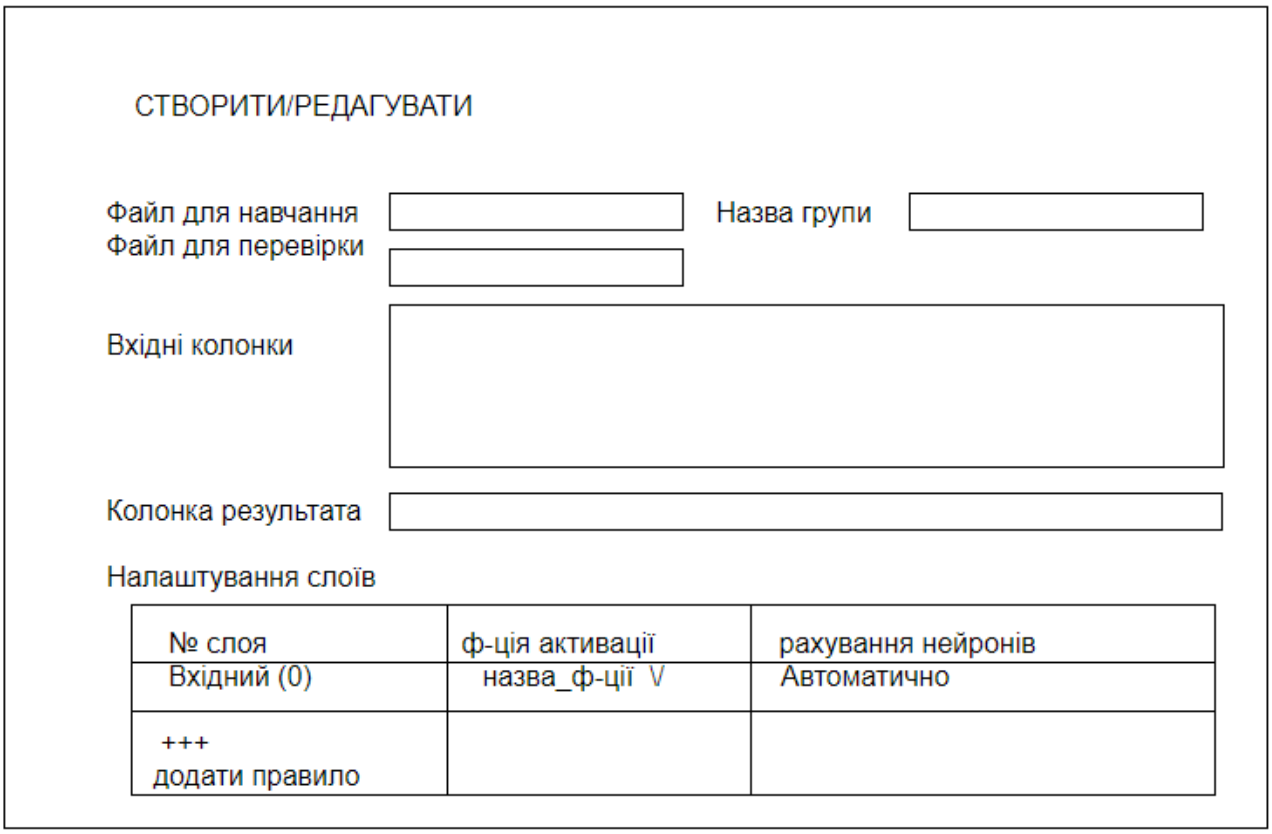


Рисунок 4.26. Створення чи редагування проєкту

Дана сторінка надає користувачу доступ до функціонала створення нового проєкту або редагування того, що вже існує. Відмінність лише в тому, що у випадку створення поля пусті з початку, а при редагуванні там вже є дані.

Потрібні елементи на сторінці:

- Поле для завантаження файлу для навчання – таблиця формату *csv, що містить у собі поля для тренування нейромережі.
- Поле для завантаження файлу для тестування – також таблиця формату *csv, з якою буде тестуватися вихідний файл
- Поле для вибору вхідних колонок – користувач вводить назви колонок із першої таблиці, які будуть вхідними даними
- Поля для вибору результуючої колонки – користувач обирає одну колонку для результату
- Налаштування шарів, що використовуються для навчання нейромережі:

- Номер шару та його назва
- Функція активації – випадаючий список з функцією вибору функції
- Метод рахування нейронів

- Додавання нового шару

3. Сторінка «Статистика»

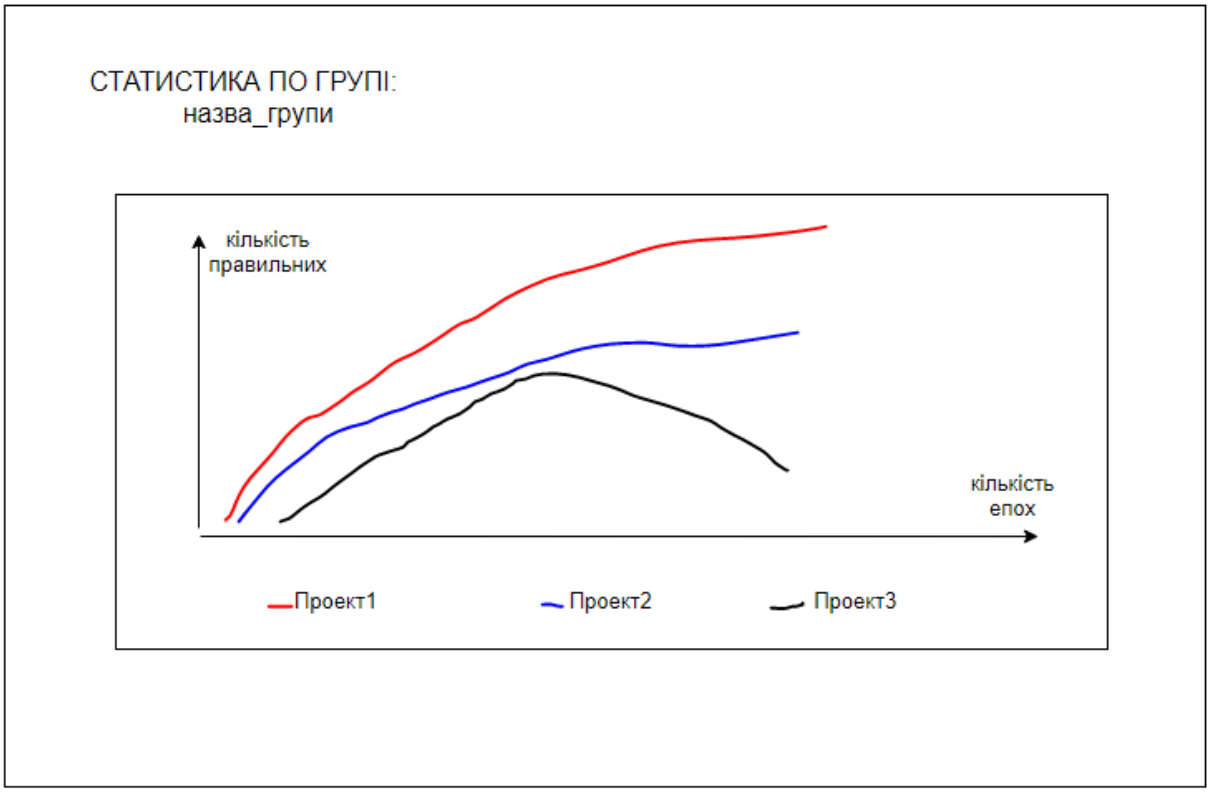


Рисунок 4.27. Перегляд статистики

По групі обраного проєкту відображається статистика у виді графіка з відношенням кількості успішних ітерацій до кількості прожитих нейромережею епох. Вона збирається з різних проєктів даної групи.

ВИСНОВКИ ДО РОЗДІЛУ 4

За основу дизайну було взято переглянуті популярні веб-сторінки, проаналізовано їх сильні та слабкі сторони, та по їхньому натхненню зроблено свій дизайн.

Так, з’ясовано, що більшість сайтів використовує кругловаті кнопочки та статичне меню. Ці знання було використано при написанні сайту. Також було досліджено методи авторизації користувачів, та обрано оптимальний для зручності і розробників і користувачів. А саме, стандартний метод за допомогою пошти або імені користувача, та через 3 соціальні мережі: 2 найпопулярніші, це Google та Facebook, і одна суто для програмістів, Github. Це рішення було прийнято з урахуванням того, що спеціалізація веб-сервісу достатньо вузька, а це означає, що будь-хто не буде ним користуватися. Тому можливість прив’язати свій репозиторій або акаунт на Github до даного веб-сервісу може бути його чудовою особливістю.

РОЗДІЛ 5. РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

5.1 Підготовка до розробки веб-сторінки.

Пропонується сервіс, який реалізовує весь достатній для зручної роботи із нейромережами функціонал, та використовує мікросервісну серверну частину. Також даний сервіс дає змогу авторизуватися та зберігати проекти у свою бібліотеку, відмічати обраним, та шукати публічні проекти інших користувачів.

У сервісі передбачений тестовий користувач, бібліотека якого є публічною, та доступ по якого є у кожного, хто продовжує користуватись сервісом без реєстрації. При цьому користувачі бачать попереджувальне повідомлення, та заклик зареєструватися для зберігання своїх проєктів у особистому кабінеті.

Для сервісу передбачається 2 ролі, а саме:

- Неавторизований користувач
- Авторизований користувач

Користувач після авторизації зможе використовувати наступний функціонал: переглянути всі свої проекти, які були створені раніше, переглянути список обраного, де можуть бути як його так і публічні роботи інших користувачів; переглядати історію того, що він відкривав чи дивився раніше, та знайти публічний проект. А також створити новий, відредагувати старий, подивитися статистику по групі, та робити стандартні дії із профілем, такі як зміна персональної інформації. Також авторизований користувач може змінити аватар, та тему сайту.

В неавторизованому вигляді користувач матиме змогу використовувати всі ті ж функції, але з публічного акаунта – тобто історія, обране та проєкти будуть вже не його особисті, а всіх людей, що користувалися акаунтом. Також користувач не матиме змогу змінювати

аватар, персональну інформацію та тему сайту.

Тобто по суті незареєстрований користувач, він же гість – це звичайний користувач, з обмеженим доступом в рамках уникнення проблеми синхронізації, та постійних змін загальнодоступного профілю.

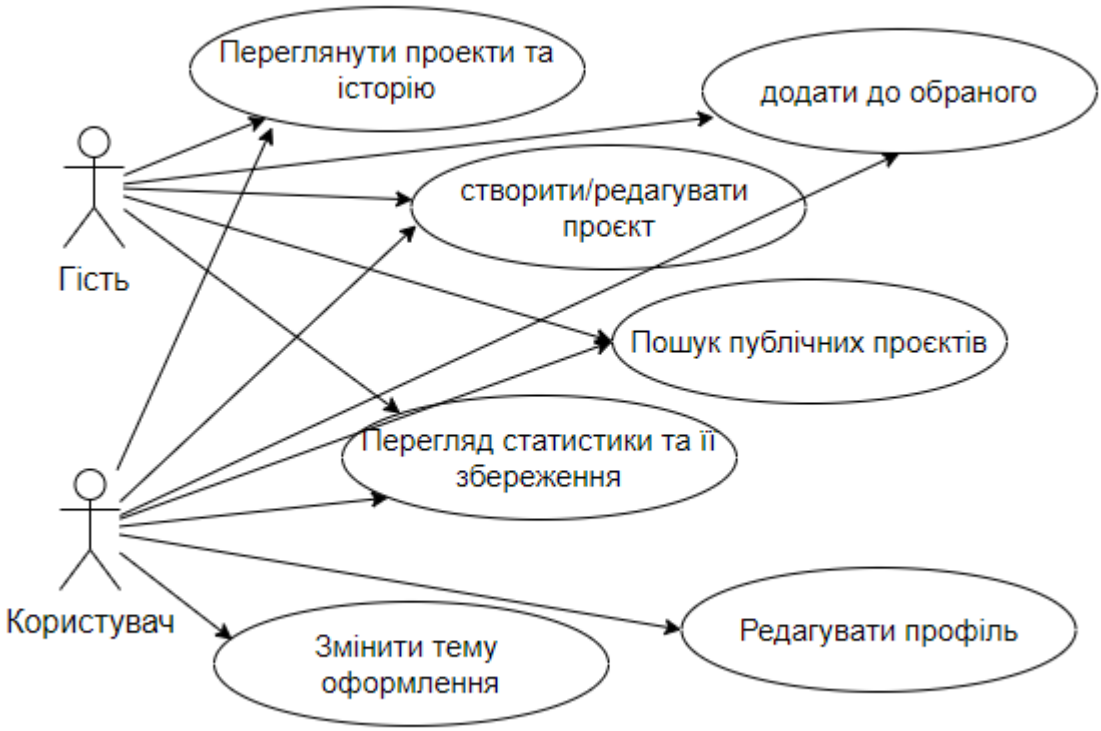


Рисунок 5.1. UseCase діаграма

Розроблюючи сайт на мовах HTML та CSS, достатньо знати основу та десяток команд для створення хорошого дизайну.

Під час написання коду на CSS було використано класи, з заданням параметрів елементам. Наприклад,

```
body{
    margin: 0;
    padding: 0;
    background: #fcfffb;
    text-align: center;
    font-family: 'Nougatine';
```

}

Означає, що у всього тіла сайту, тобто у кожного елемента веб-сторінки будуть нульові відступи, фон заданого кольору, та розташування тексту по центру. Це налаштування за замовчуванням, але якщо у якогось елемента будуть свої налаштування, то вони стануть більш пріоритетними.

Також використовуються операції при певних діях.

@media (max-width: 600px) {

.menu-main li {display: block;}}

Тут при ширині менше вказаної, тобто 600 пікселів, для класу menu-main елементу списку буде підключене відображення вертикального списку. При тому, що зазвичай воно горизонтальне. Це робиться для того, щоб при збільшенні масштабу користувачу було зручніше обирати пункти меню, при маленькому екрані.

5.2 Розробка стартової сторінки

Тож, перейдемо до створення веб-сайту.

Починаючи з початку, маємо таку реалізація головної сторінки:

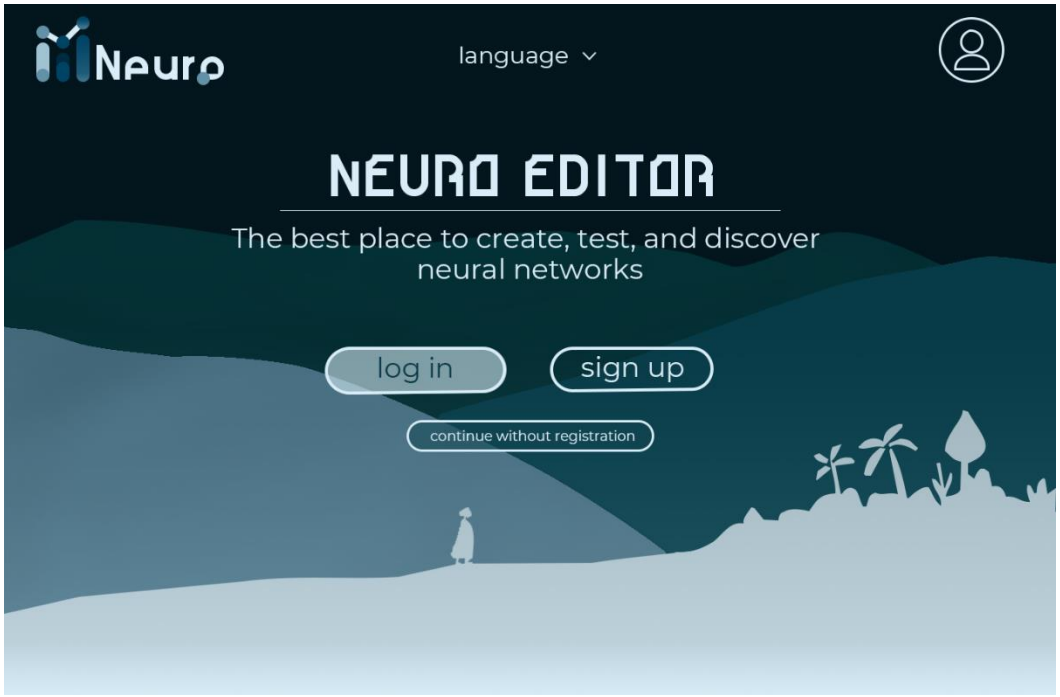


Рисунок 5.2 Стартова сторінка

Як і планувалося, на головній сторінці є фонове зображення, текст щодо того, що з себе представляє веб-сторінка і варіанти входу в систему.

Малюнок намальований власноруч, і виконаний як накладання п’яти шарів. У наступному оновленні можна зробити так, щоб при перемотуванні вони ховалися один за одним.

Розглянемо шапку детальніше.

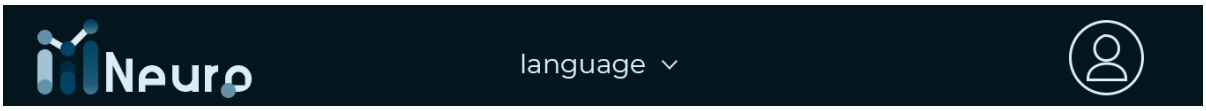


Рисунок 5.3. Шапка сайта на стартовій сторінці

Бачимо мінімальний набір функціонала: логотип з можливістю переходу на головну сторінку, функціонал вибору мови, реалізований як випадаючий список, та іконку логотипу, натиснувши на яку теж можна зареєструватися або увійти в систему.

1. Логотип

Рисунок 5.4. Кольоровий варіант логотипу

На всіх інших сторінках, окрім стартової, використовується кольоровий варіант логотипу.

За його основу при створенні було взято зв’язок нейронів між собою (верхня частина) та можливість вмикання/вимикання наборів функцій, реалізовані як перемикачі (нижня частина). Логотип виконаний в підібраній кольоровій гамі. Підібраний такий шрифт, щоб підкреслити здатність людського мозку розрізняти літери, навіть коли вони не мають половини своєї форми. Відтворення аналогічної мозкової діяльності і є метою даної дипломної роботи.

2. Вибір мови сайту

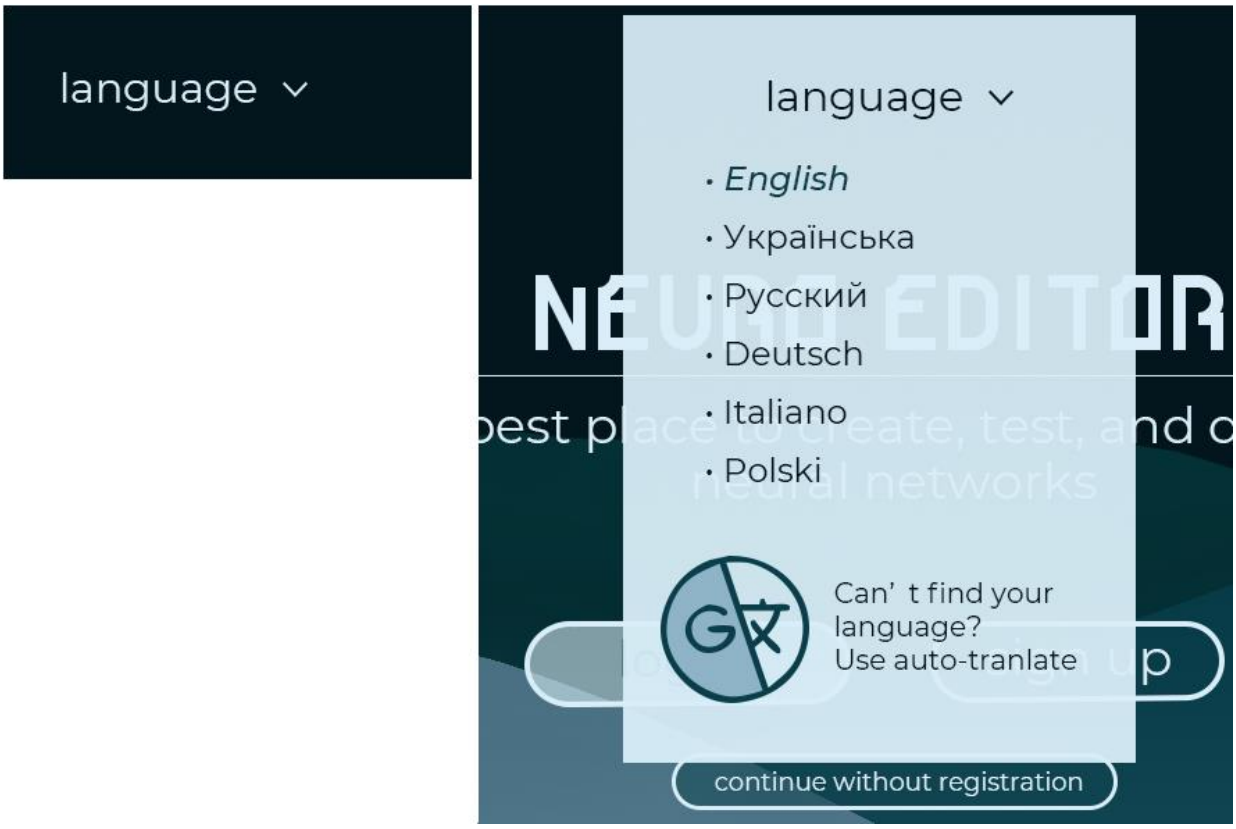


Рисунок 5.5. Поле для вибору мови (зліва), та варіанти мов (справа)

Функція зміни мови реалізована як випадаючий список. В ньому є поширені мови, та наша рідна – українська. Обрана мова має вигляд напівжирного курсиву.

Оскільки неможливо передбачити всі мови для розробки сайту, то для користувачів із мовами, що не ввійшли в список, є функція автоматичного перекладу за допомогою Google Translate. Звичайно, такий переклад не буде ідеальним, але допоможе, якщо людина не знає жодної із перерахованих мов.

Також є велика іконка Перекладача, щоб навіть без змоги прочитати текст, можна було зрозуміти, що це означає.

3. Поле для авторизації

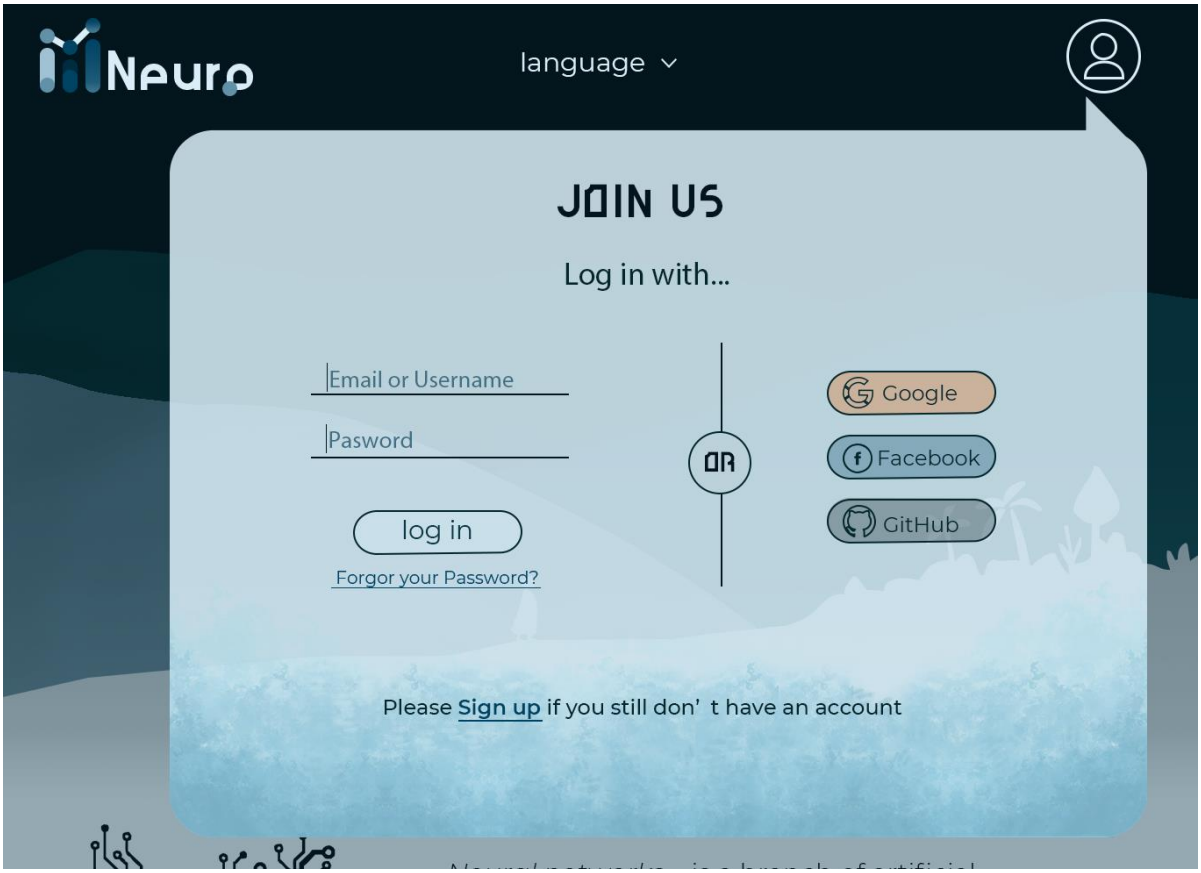


Рисунок 5.6. Поле для входу користувача в систему

На даній веб-сторінці реалізовано функціонал входу в систему, використовуючи різні методи.

Перший – вхід через електронну пошту або ім’я користувача.

Також тут присутня функція відновлення паролю

Другий – вхід через соціальні мережі. Оскільки зараз люди мають акаунти в багатьох мережах, то зручніше використовувати логін та пароль, які вже знаходяться в пам’яті, людській чи комп’ютерній, протягом певного часу. Використано 3 популярні мережі: Google, Facebook, та Github.

Оскільки веб-сервіс орієнтовано на програмістів, то можливість використовувати свій репозиторій буде великою перевагою.

Також знизу є повідомлення, про те що робити користувачу, який не зареєстрований. При натисканні на посилання відбудеться перенаправлення на

сторінку реєстрації.

4. Поле профілю авторизованого користувача

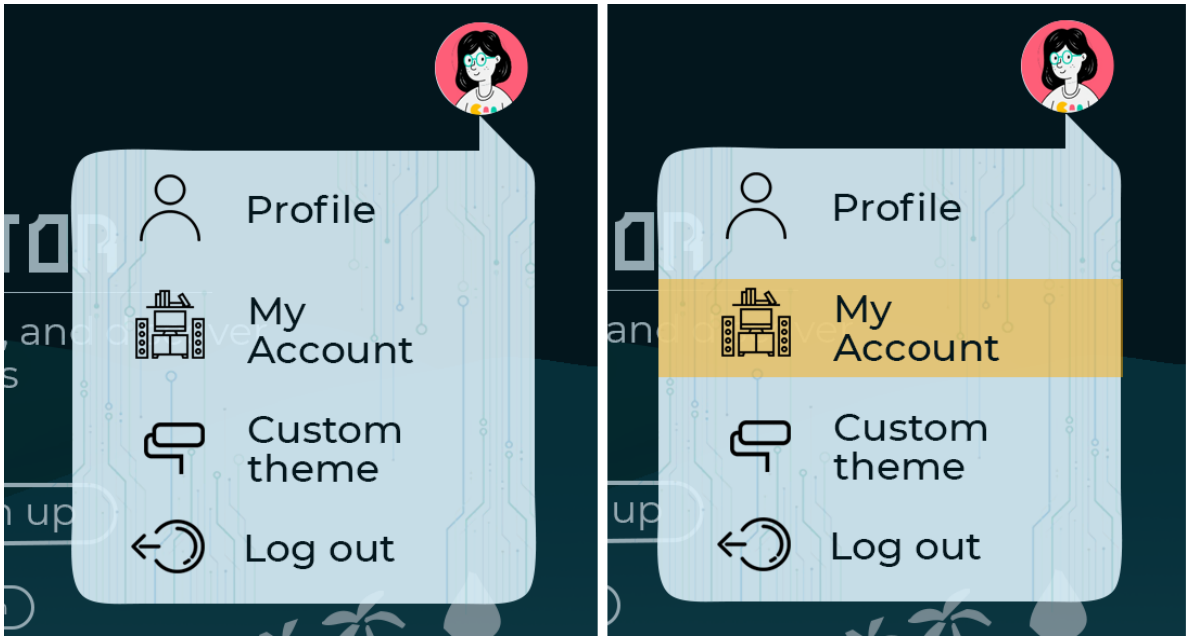


Рисунок 5.7. Перегляд кнопки профілю із шапки (зліва) та зміна поля при наведенні на нього курсору (зліва)

При натисканні на іконку профілю у авторизованого користувача можемо здійснити перехід до Профілю, аккаунта, налаштування теми, та функція виходу із свого облікового запису.

У авторизованого користувача вигляд шапки відрізняється.

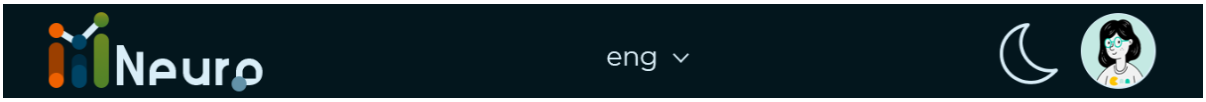


Рисунок 5.8. Шапка авторизованого користувача

Бачимо кольоровий логотип, обрану мову, обраний аватар, та кнопка налаштування теми сайту.

З шапкою розібралися, далі подивимося на те, що нижче на стартовій сторінці.

Під областю авторизації з фоновим зображенням іде короткий опис проблематики веб-сервісу, щоб користувач міг зрозуміти, навіщо він тут, і чому йому варто використати даний веб-сервіс.

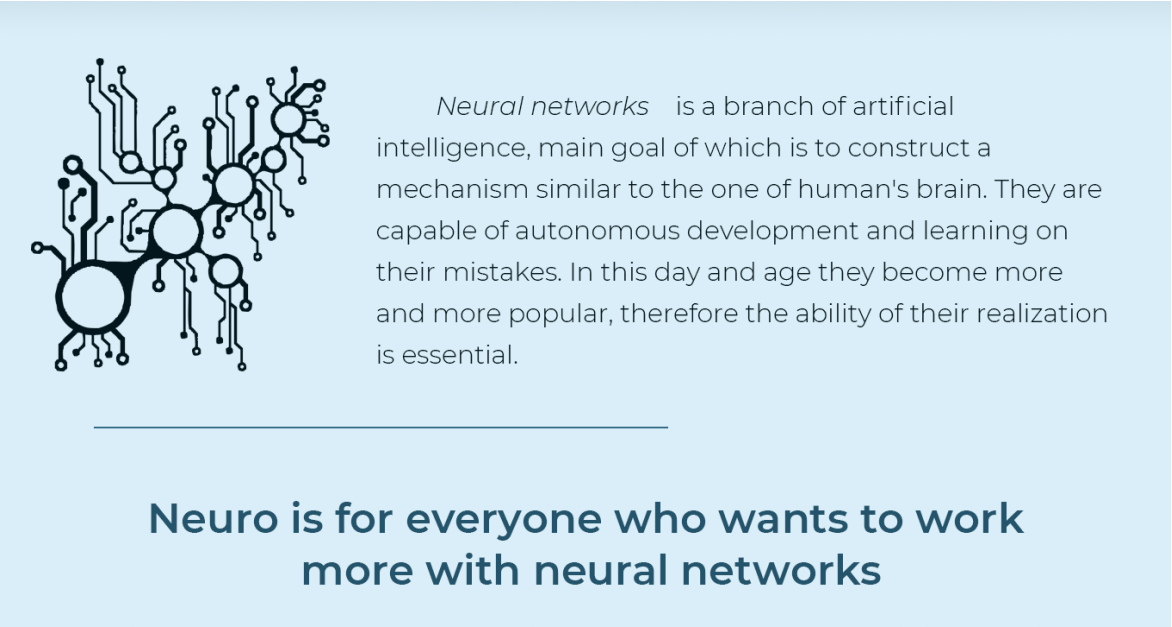


Рисунок 5.9. Загальний текст відомості про веб-сервіс

Під даним текстом розташовано висловлювання, стосовно заклику користувачів використовувати даний веб-сервіс.

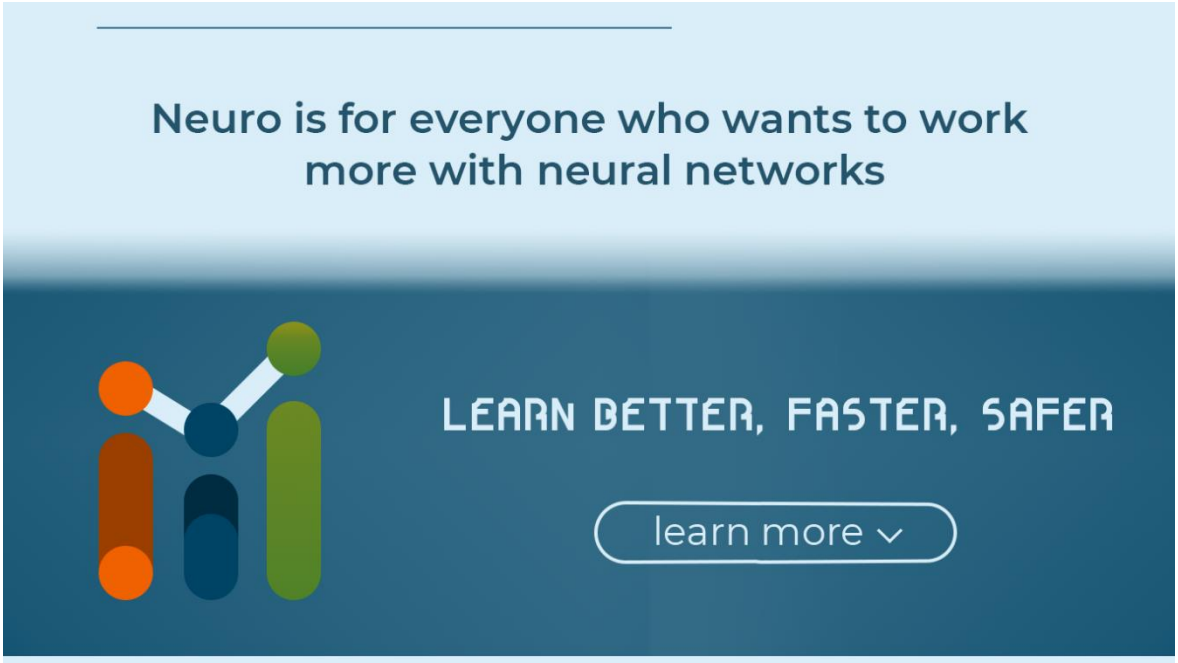


Рисунок 5.10. Поле детальнішої інформації

Нижче приведено розділ з перевагами розробленого веб-сервісу, в якому при натисканні на кнопку поле розгортається:

Рисунок 5.11. Відкрите поле детальнішої інформації

Присутнє пояснення Гасла, та статистика по роботі веб-сервісу. І знову заклик приєднатися, після того як користувач вже прочитав переваги, і переконався чому йому варто використовувати веб-сервіс.

Знизу стартової сторінки, як і на всіх інших, розміщено розділ з копірайтом

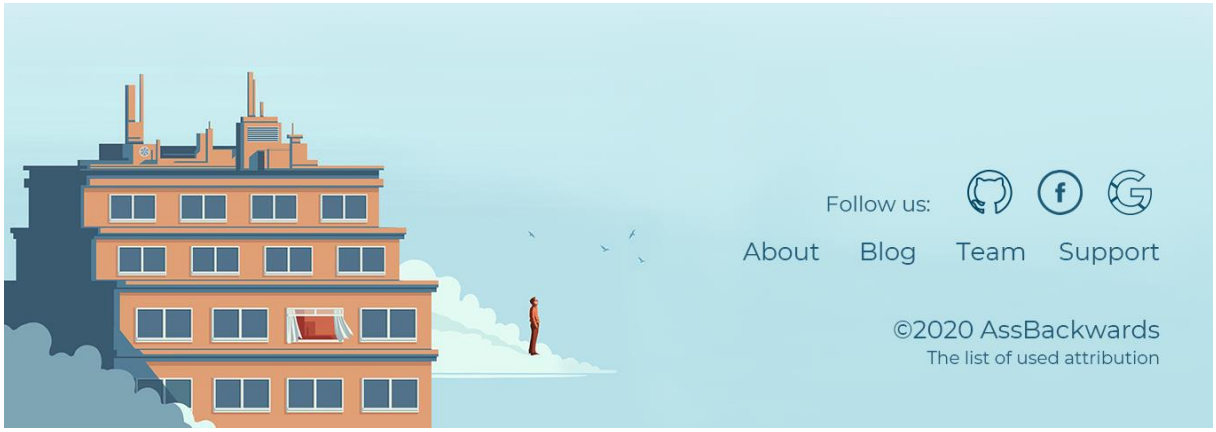


Рисунок 5.12. Поле копірайту [7]

Це поле дає змогу користувачу вивчити більше інформації про веб-сервіс, ознайомитися із командою розробників, а також почати слідкувати за новинами у соціальних мережах або переглянути репозиторій на Github.

Копірайт – це важливо, отже це поле присутнє на всіх сторінках.

5.3 Сторінка Профілю

Нехай користувач зареєструвався, і тепер хоче перейти на сторінку профіля. Зробивши це, він побачить вказане на рисунку нижче (рисунок 5.13).

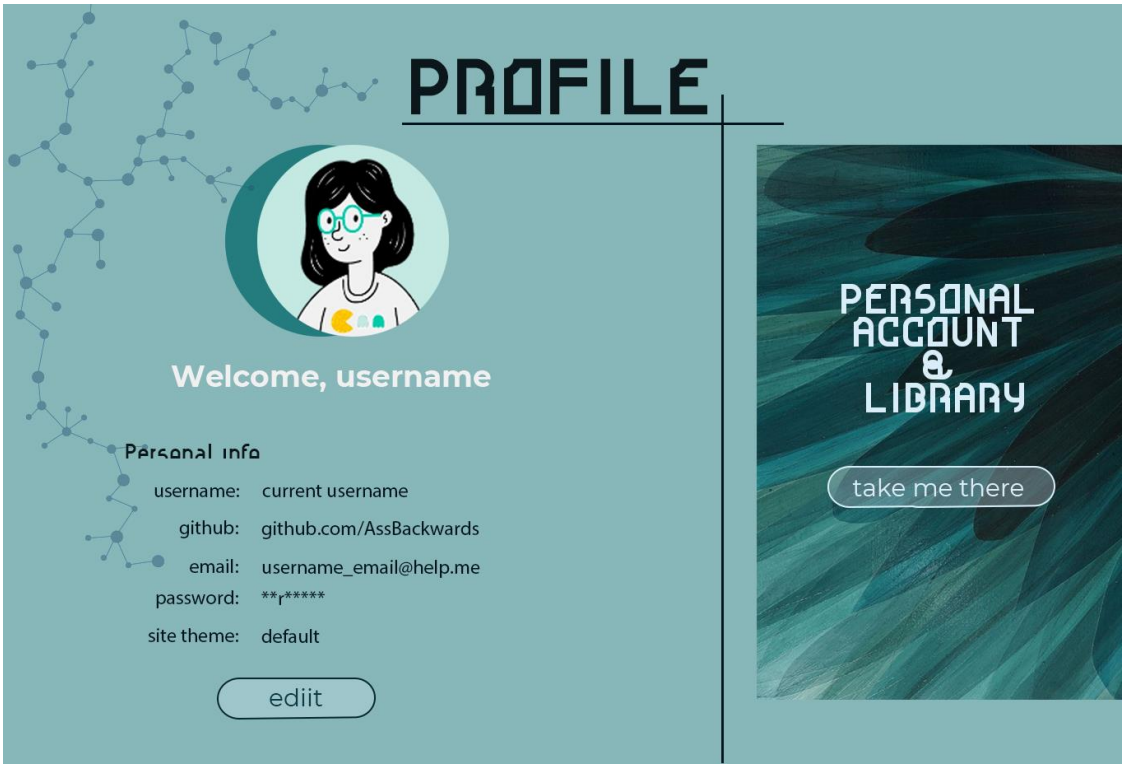


Рисунок 5.13. Профіль користувача [8]

Наведена інформація профілю, та кнопка редагування.

Інформація складається з наступних елементів: фото, ім’я, посилання на акаунт в Github, пошта, пароль, та обрана тема оформлення.

Збоку реалізовано поле для швидкого переходу в особистий кабінет, у якому зберігаються всі проєкти користувача, та до цього ще повернемося.

При виборі редагування профілю відкривається така сторінка:

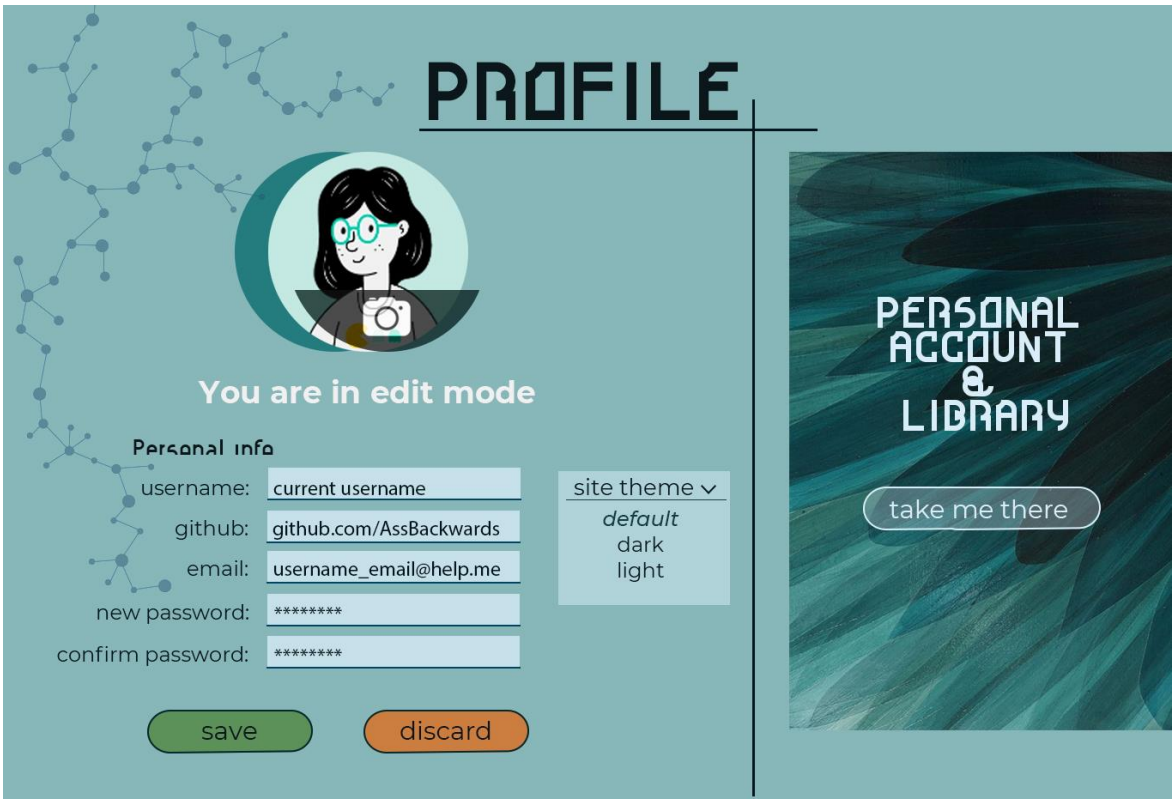


Рисунок 5.14. Редагування профілю [8]

Присутня можливість редагування всіх елементів профілю, випадаючий список для вибору теми, та подвійне поле введення паролю для його підтвердження.

Важливою функцією є можливість як зберегти, так і скасувати внесені зміни.

Для обрання зміни аватара профілю розроблено таке поле:

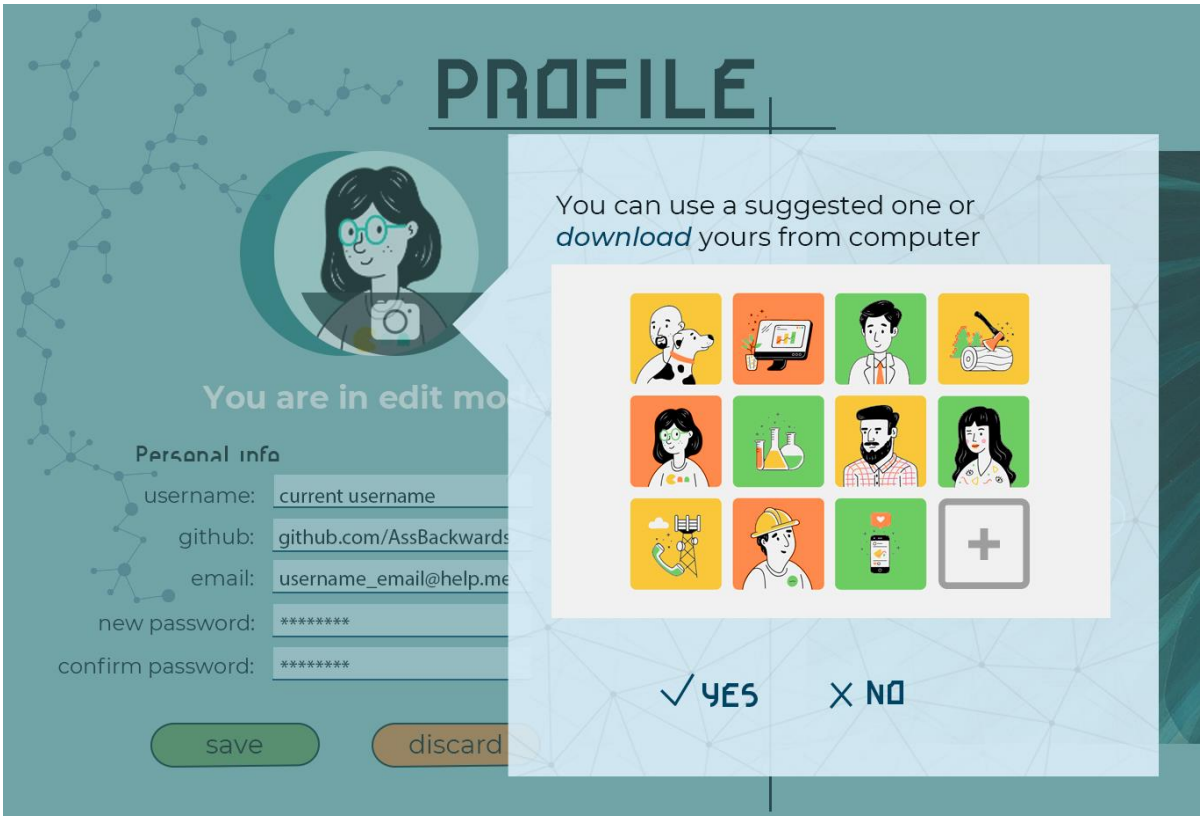


Рисунок 5.15. Зміна аватару профілю [7]

Якщо є бажання зареєструватися на сайті, користувач потрапляє на наступне вікно вказане у рисунку 5.16.

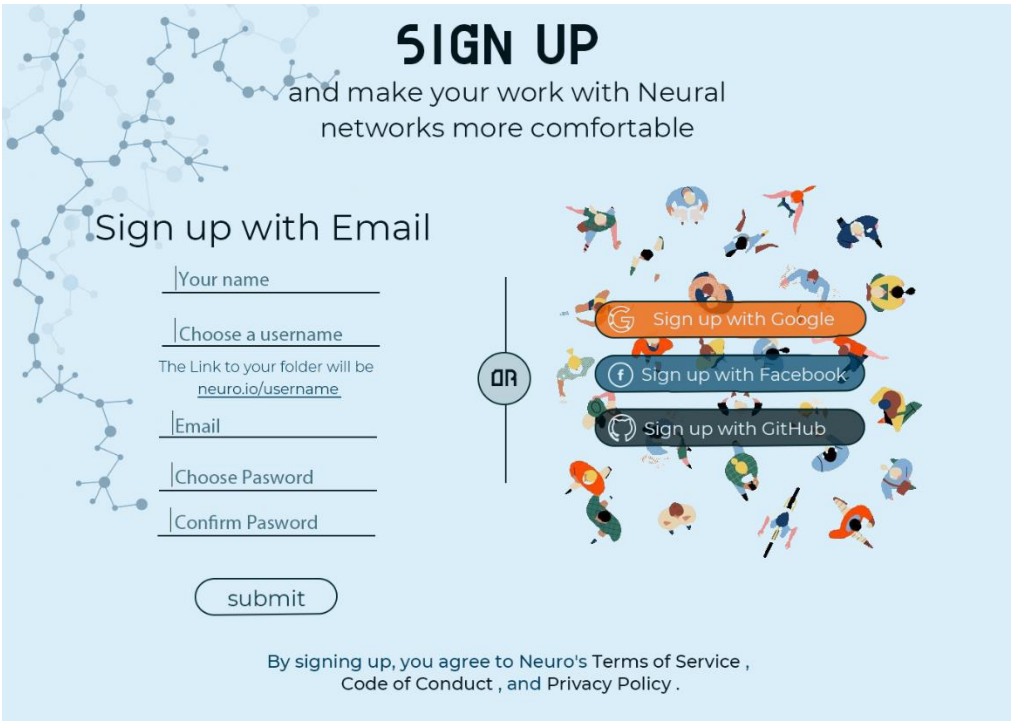


Рисунок 5.16. Реєстрація нового користувача

Сервісом передбачена реєстрація як через пошту, вказавши нікнейм, так і через соціальні мережі. Загалом сторінка реєстрації схожа на сторінку входу в акаунт, просто розширена такими полями як Ім’я та підтвердження пароля.

5.4 Особистий кабінет користувача

При переході в особистий кабінет реалізовано наступний інтерфейс:

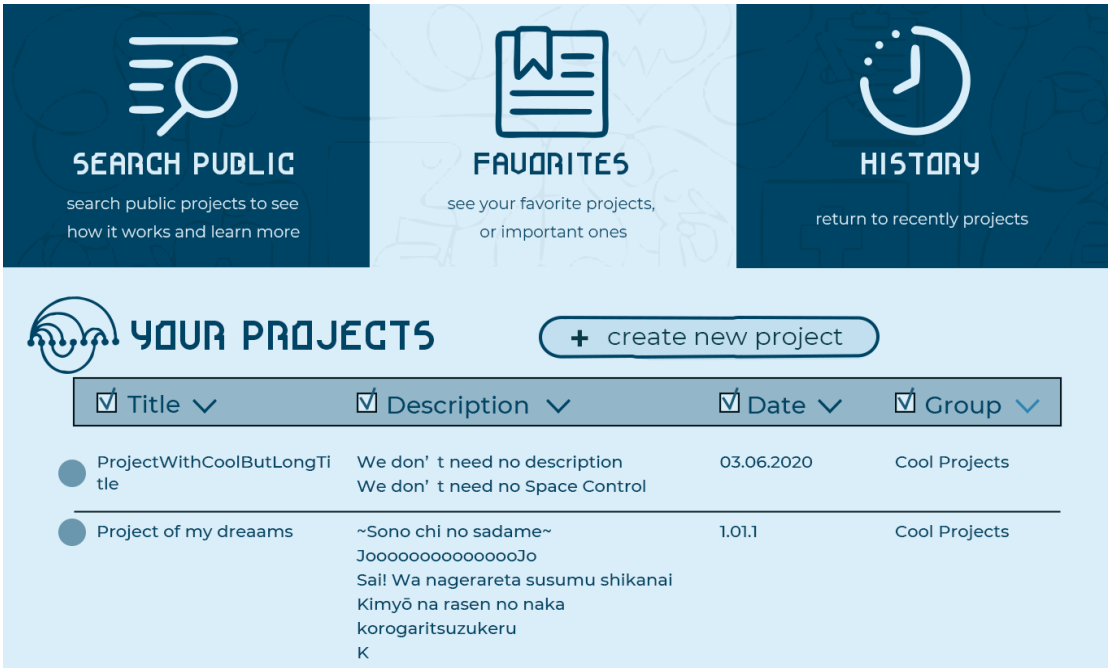


Рисунок 5.17. Сторінка персонального кабінету

Зверху 3 категорії:

- Історія

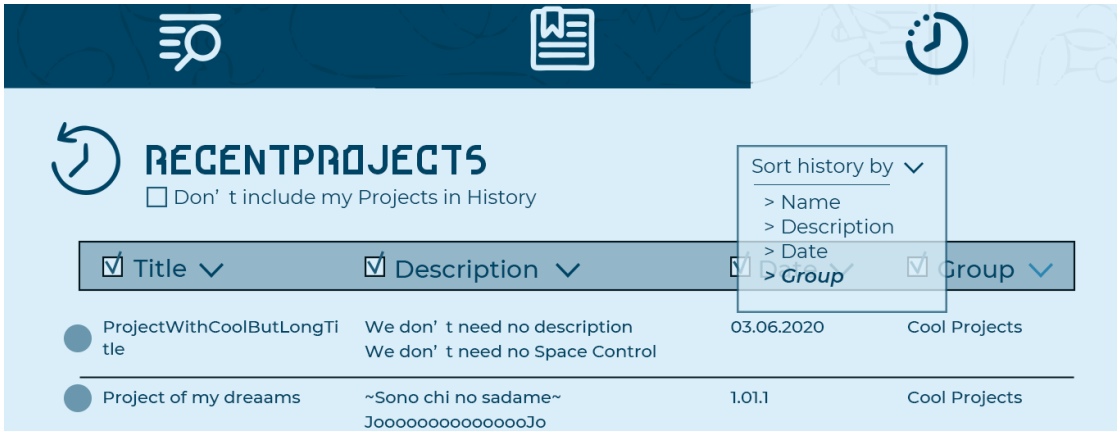


Рисунок 5.18. Історія переглянутих проєктів.

Обираючи один із трьох верхніх розділів, обране підсвічується білим, а весь розділ зменшується у висоті.

В даному розділі користувач може подивитися історію відкритих та переглянутих проєктів.

Також можна не включати в історію свої проекти – коли потрібно переглянути лише публічні. Присутній функціонал сортування історії.

- Улюблене

Рисунок 5.19. Обрані проєкти.

Даний розділ дає користувачу змогу зберігати свої або публічні проєкти до обраного. Тобто тут можна зберігати важливі, ті, які сподобалися, та просто цікаві проєкти. Також можна не показувати свої проєкти у списку обраного.

- Пошук

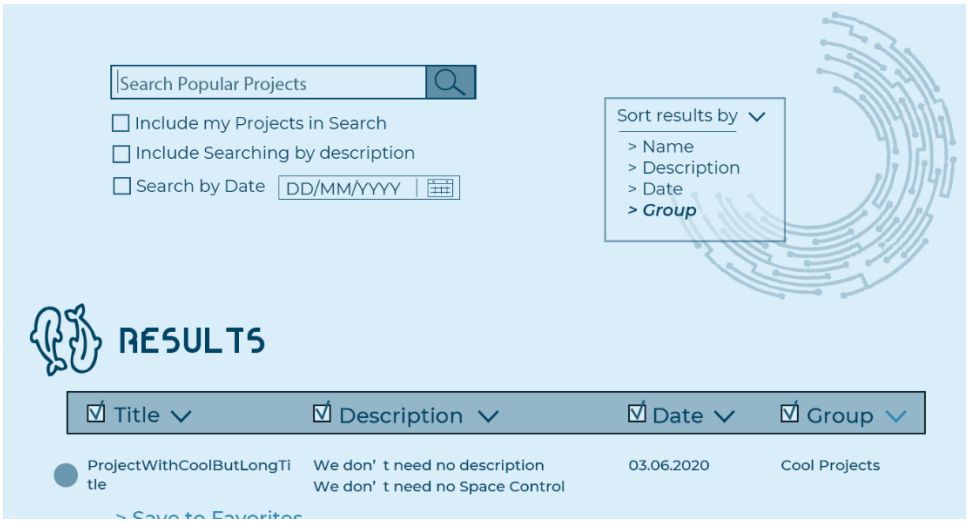


Рисунок 5.20. Вкладка пошуку

Користувач може здійснювати пошук проєктів за їх назвою, назвою групи, відображаючи результати відсортованими по основних параметрах.

Є можливість не показувати свої проєкти при пошуку, а також здійснювати пошук за описом.

Сама ж вкладка проєктів має наступний вигляд:

Рисунок 5.21. Проєкти користувача

Переглядаючи створені проєкти, користувач може приховати один із параметрів показу, наприклад, групу, знявши галочку.

Кольорові кола – як індикатор групи, для зручності.

Щоб отримати доступ до статистики чи поля редагування проєкту, користувач повинен натиснути на обраний, і з’явиться таке поле як на рисунку 5.22. Також на рисунку видно, що проєкти відсортовані по групі:

При виборі функції відредагувати проєкт, відкривається те ж поле що і при створенні. Єдина відмінність у тому, що при редагуванні дані вже внесені, а при створенні всі поля пусті.

Рисунок 5.22. Список проєктів, відсортованих по групі

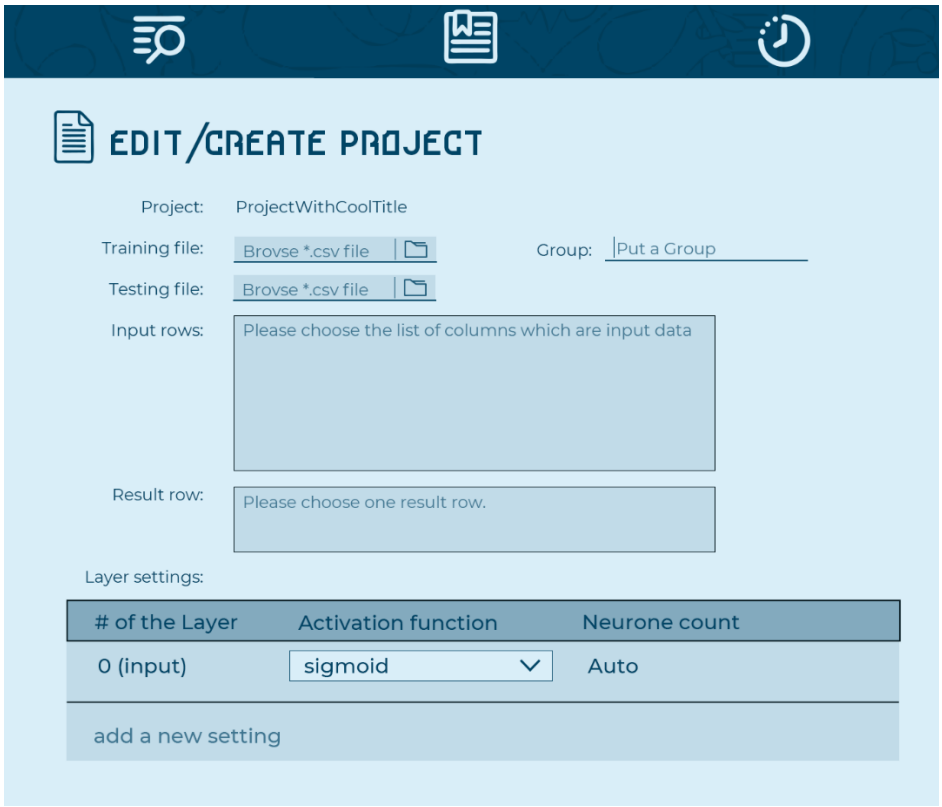


Рисунок 5.23. Редагування/Створення проєкту

Поле статистики виглядає аналогічним чином. Бачимо це на рисунку 5.23.

Також користувач має змогу зберегти отриманий графік на комп’ютер, доступні параметри: *.pdf, *.png. Формат таблиці недоступний через складність реалізації, а *.jpg через сильне стиснення файлу.

У вікні статистики можемо спостерігати графік по групі, із вказаними кольорами кривих та відповідних їм проєктів.

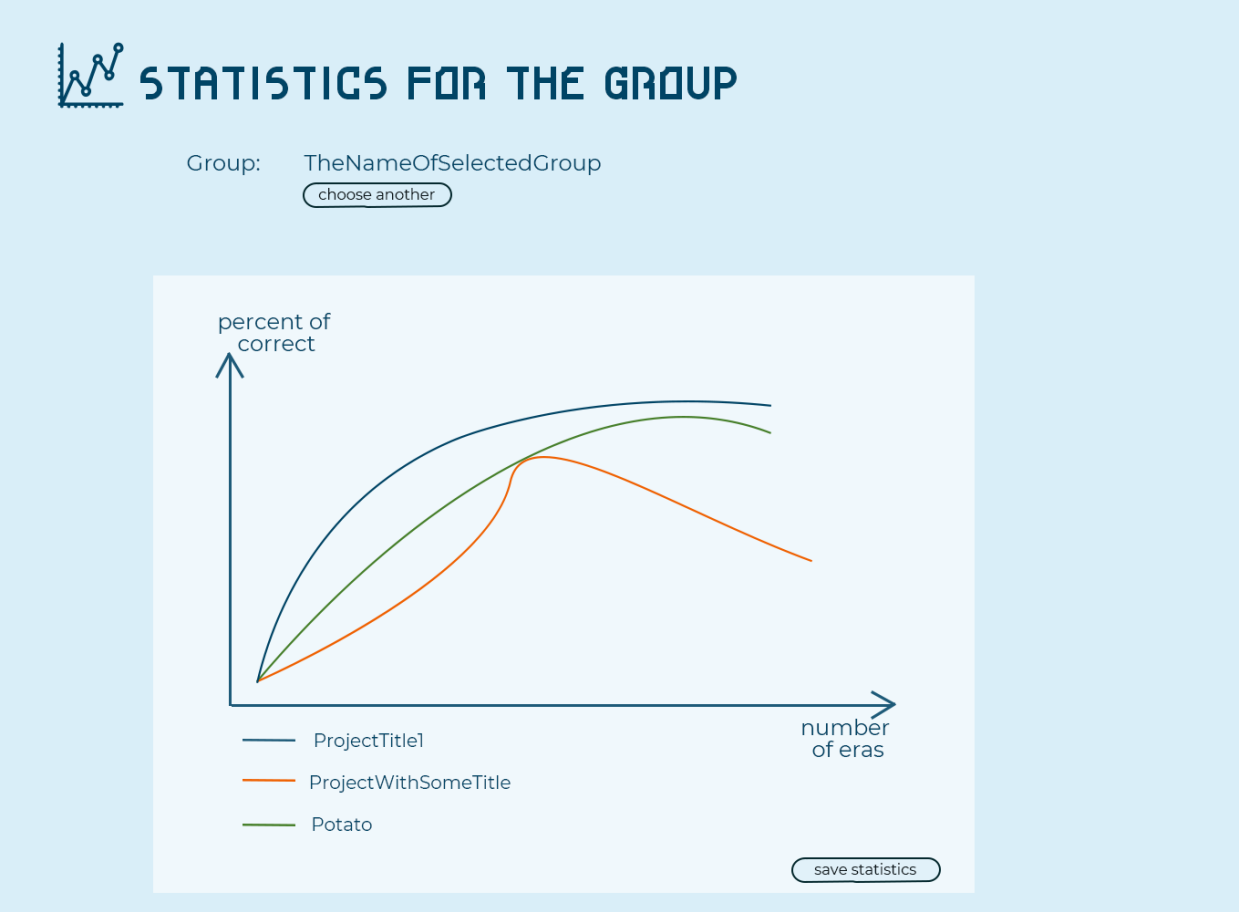


Рисунок 5.24. Статистика по групі нейромереж

Оскільки для сайту було використано запозичені матеріали, то всі посилання прикріплені в окремому вікні, названому “Atributes”.

Рисунок 5.25. верх сторінки з посиланнями

Як можна бачити, сторінка з посиланнями стилізована під весь сайт.

Під логотипом бачимо наступний перелік посилань:

Рисунок 5.26. Посилання на ресурси.

Також було розроблено темну тему для сайту, яку продемонстровано у даній роботі на прикладі профілю (рисунок 5.27).

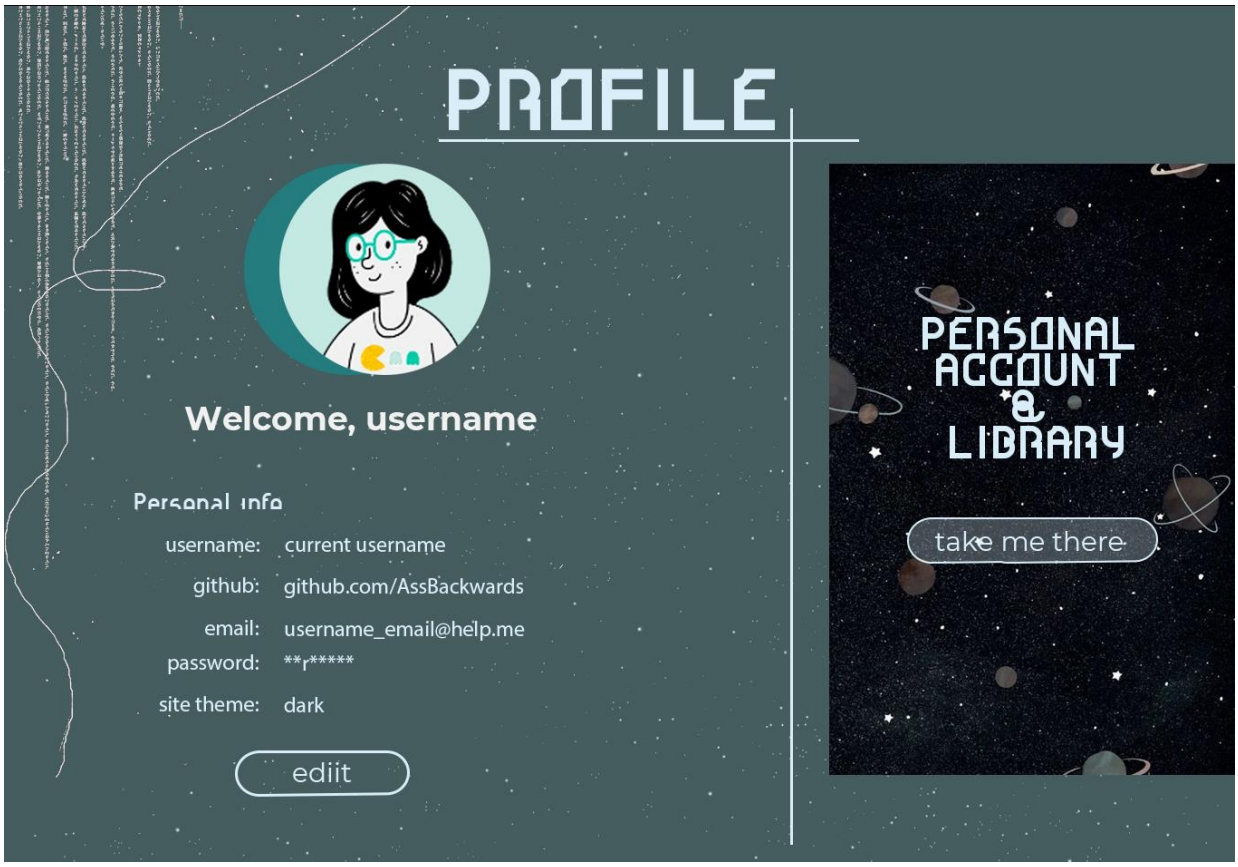


Рисунок 5.27. Темна тема для роботи вночі [8]

Розділ копірайту матиме вигляд не світло-блакитного будиночка, а човника у темному морі вночі (рисунок 5.28).

Рисунок 5.28. Темна тема копірайта [8]

А загальний фон сайту замість блакитно-білого набуватиме вигляд неба із зорями (рисунок 5.29).

Рисунок 5.29. Загальний фон

5.5 Дизайн системних повідомлень

Декілька системних повідомлень:

Рисунок 5.30. Помилка 504 [8]

Рисунок 5.31. Помилка 404 [8]

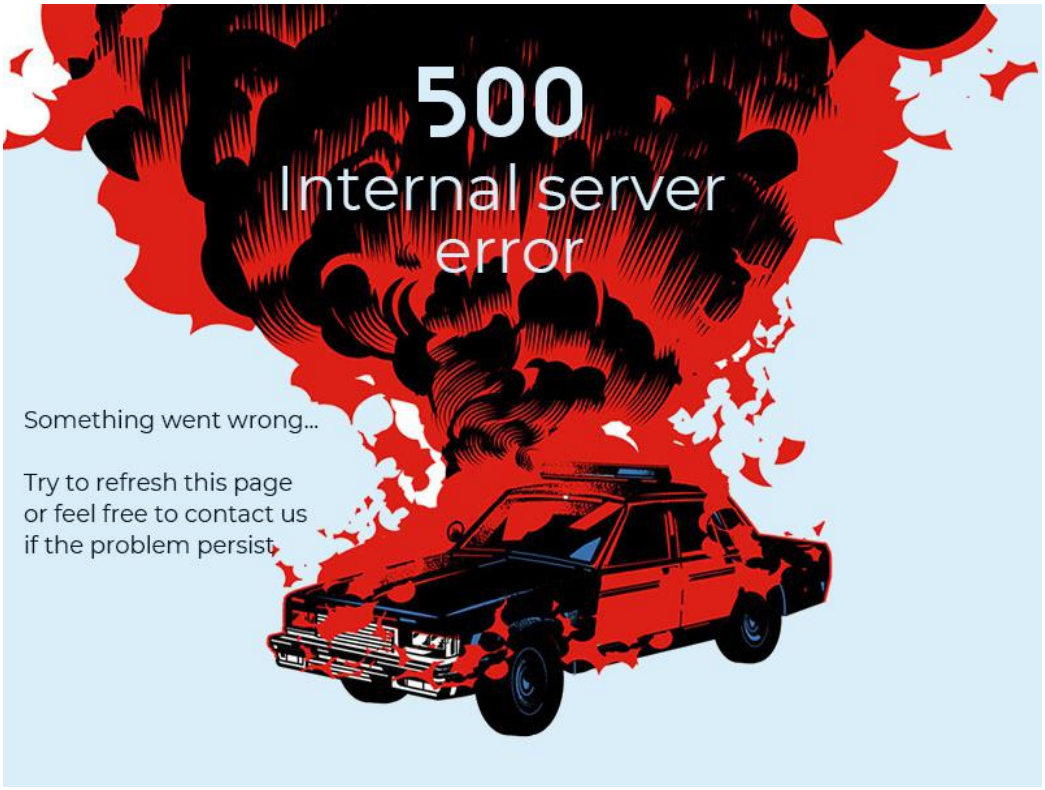


Рисунок 5.32. Помилка 500 [7]

Також розроблено системне повідомлення про вхід в систему, на випадок якщо автоматична переадресація не відбудеться.

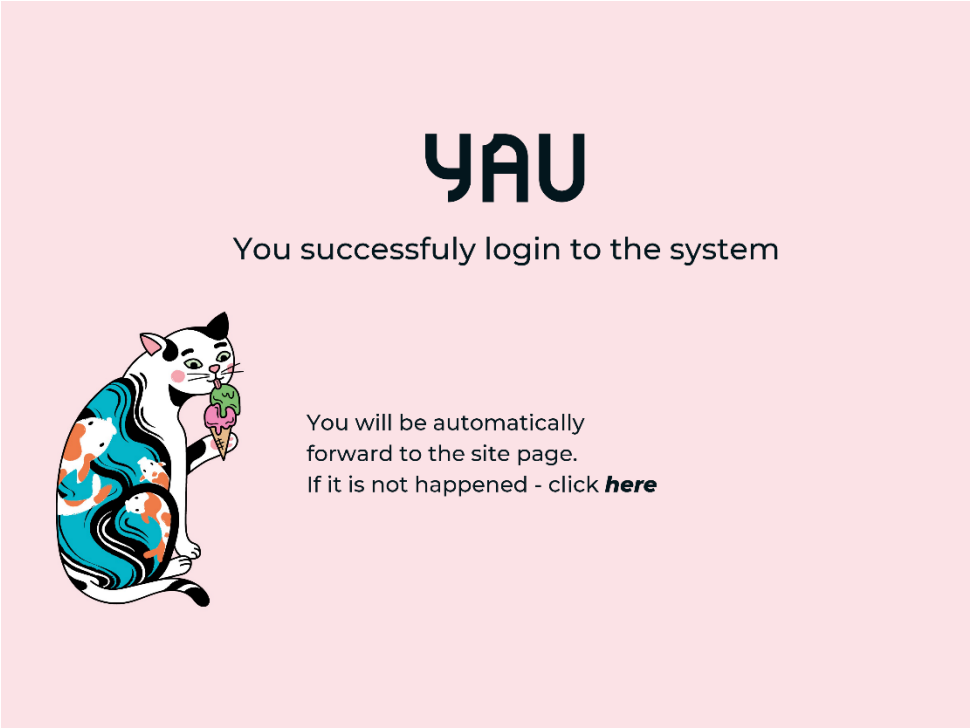


Рисунок 5.33. Повідомлення про вхід в систему [7]

ВИСНОВКИ ДО РОЗДІЛУ 4

Створено веб-сервіс на популярних мовах HTML, CSS, та Javascript, а саме фреймворк JQuery.

HTML та CSS – чудове рішення для написання веб-додатку, і допомагають створити сучасні динамічні сайти. Протягом багатьох років HTML удосконалювався різними розробниками, щоб отримати в результаті те що є. Можна створити зручний багатофункціональний сайт використовуючи лише HTML у комбінації з CSS, а якщо додати до цього Javascript, то отримаємо чудовий сайт, автоматизований для будь-якого пристрою, від маленького мобільного екрана, до великого комп'ютерного.

Використовуючи розроблений раніше макет та функціональність використаних мов було розроблено сайт, який має клієнтоорієнтованість на високому рівні.

Було підібрано як кольори для сайту, так і малюнки до цих кольорів. Також, враховуючи спеціалізацію сайту, було додано ряд особливих функцій, таких як реєстрація за допомогою Github.

Отриманий сайт в цілому приємний на вигляд, та зрозумілий у користуванні.

ЗАГАЛЬНІ ВИСНОВКИ

Під час виконання дипломного проекту був розроблений веб-сервіс, з використанням мікросервісів для роботи із нейромережами широкого призначення, клієнтська частина, тобто фронтенд.

Використано мови HTML, CSS, Javascript, оскільки аналогів їм практично немає, для написання сайту з нуля. Оскільки комбінацією цих мов можна реалізувати майже все, що приходить в голову, то розробка сайту не є надскладною задачею.

Отриманий сервіс цілком реалізовує функціонал, необхідний для коректної роботи бекенду, навіть має декілька додаткових функцій, таких як окремий профіль, темна тема, зміна мови і так далі.

В процесі розробки проєкту, було проаналізовано теоретичні відомості про мікросервіси, фронтенд. Мікросервісний фронтенд дуже корисний коли проєкт настільки розрісся, що підтримка одного монолітного коду важче, ніж переписати його на мікросервіси. Існують компанії, успішні у даному перевтіленні архітектури, однак всі вони вже були великими на той момент, і їх підтримували команди із компетентних розробників.

Для розв'язання проблеми, піднятої в рамках даної дипломної роботи, немає потреби реалізовувати мікрофронтенд, оскільки це було б зайвим навантаженням на код програми.

ПЕРЕЛІК ПОСИЛАНЬ

1. TheNewStack [Електронний ресурс] – Режим доступу до ресурсу: https://thenewstack.io/led-amazon-microservices-architecture/
2. Runscope Blog [Електронний ресурс] – Режим доступу до ресурсу: https://blog.runscope.com/posts/monolith-microservices-transforming-real-world-ecommerce-platform-using-strangler-pattern?author=58ec33a515d5db8a63ffa523
3. Evaluation to Implementation: The Coca-Cola Company Journey [Електронний ресурс] – Режим доступу до ресурсу: https://www.mulesoft.com/ty/webinar/evaluation-to-implementation-coca-cola-company-journey
4. How Netflix migrated from a monolithic to a microservice architecture [Електронний ресурс] – Режим доступу до ресурсу: https://hub.packtpub.com/how-netflix-migrated-from-a-monolithic-to-a-microservice-architecture-video/
5. Microservice Architecture — Learn, Build, and Deploy Applications – Easy as Pie [Електронний ресурс] – Режим доступу до ресурсу: https://dzone.com/articles/microservice-architecture-learn-build-and-deploy-a
6. E-commerce [Електронний ресурс] – Режим доступу до ресурсу: https://www.calltouch.ru/glossary/elektronnaya-kommertsiya/
7. Maycon Prasniewski Working on the thin line between design and illustration. [Електронний ресурс] – Режим доступу до ресурсу: https://dribbble.com/mayconpras
8. Pinterest [Електронний ресурс] – Режим доступу до ресурсу: https://www.pinterest.com/pin/51298883244661996/
9. Fruit icons by icons network [Електронний ресурс] – Режим доступу до ресурсу: https://www.iconfinder.com/iconsets/fruit-102
10. PayPal [Електронний ресурс] – Режим доступу до ресурсу: https://www.paypal.com/ua/home

11. Netflix [Електронний ресурс] – Режим доступу до ресурсу: https://www.netflix.com/
12. Google Fonts [Електронний ресурс] – Режим доступу до ресурсу: https://fonts.google.com/
13. InColorBalance [Електронний ресурс] – Режим доступу до ресурсу: https://color.romanuke.com/tsvetovaya-palitra-4088/
14. Montserrat font [Електронний ресурс] – Режим доступу до ресурсу: https://fonts.google.com/specimen/Montserrat
15. Різновиди тестування Мікросервісів [Електронний ресурс] – Режим доступу до ресурсу: https://qaat.ru/raznovidnosti-testirovaniya-mikroservisov/

ДОДАТОК 1

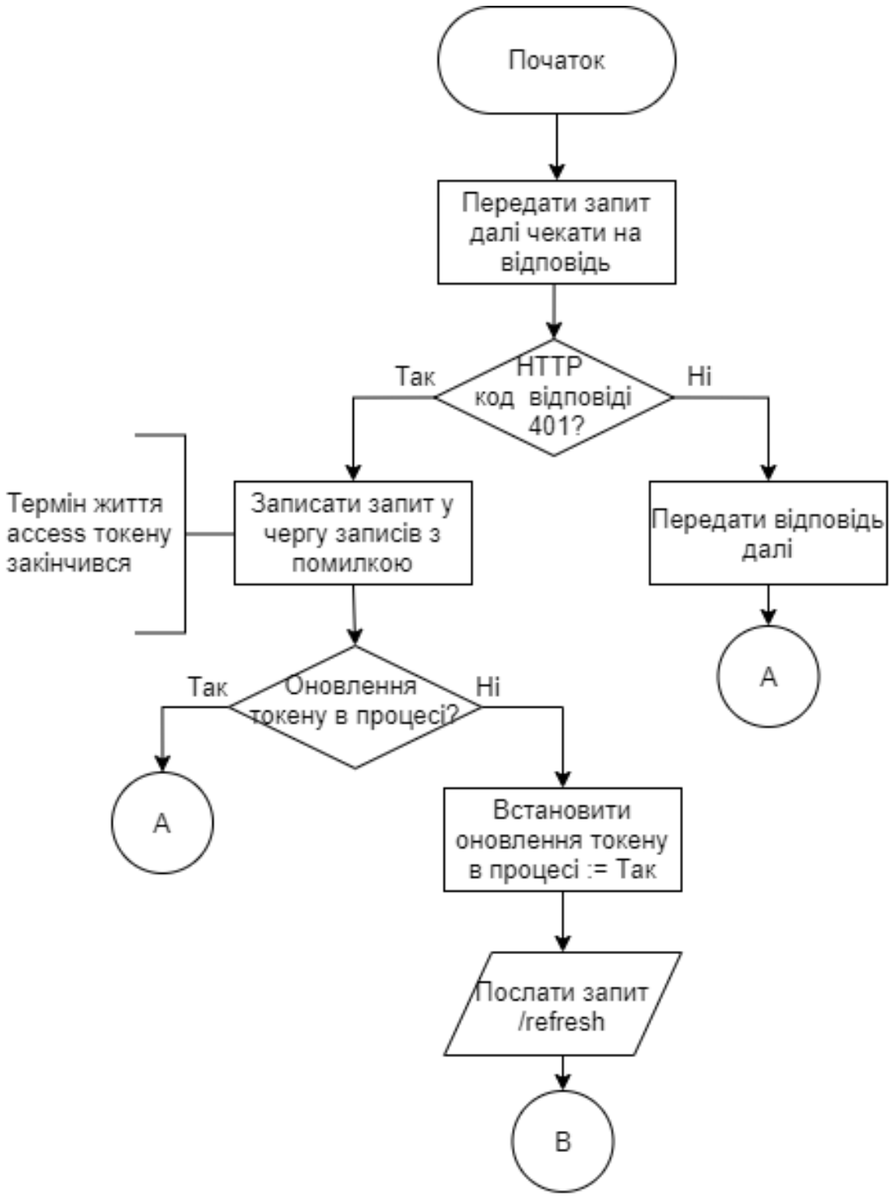
Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення

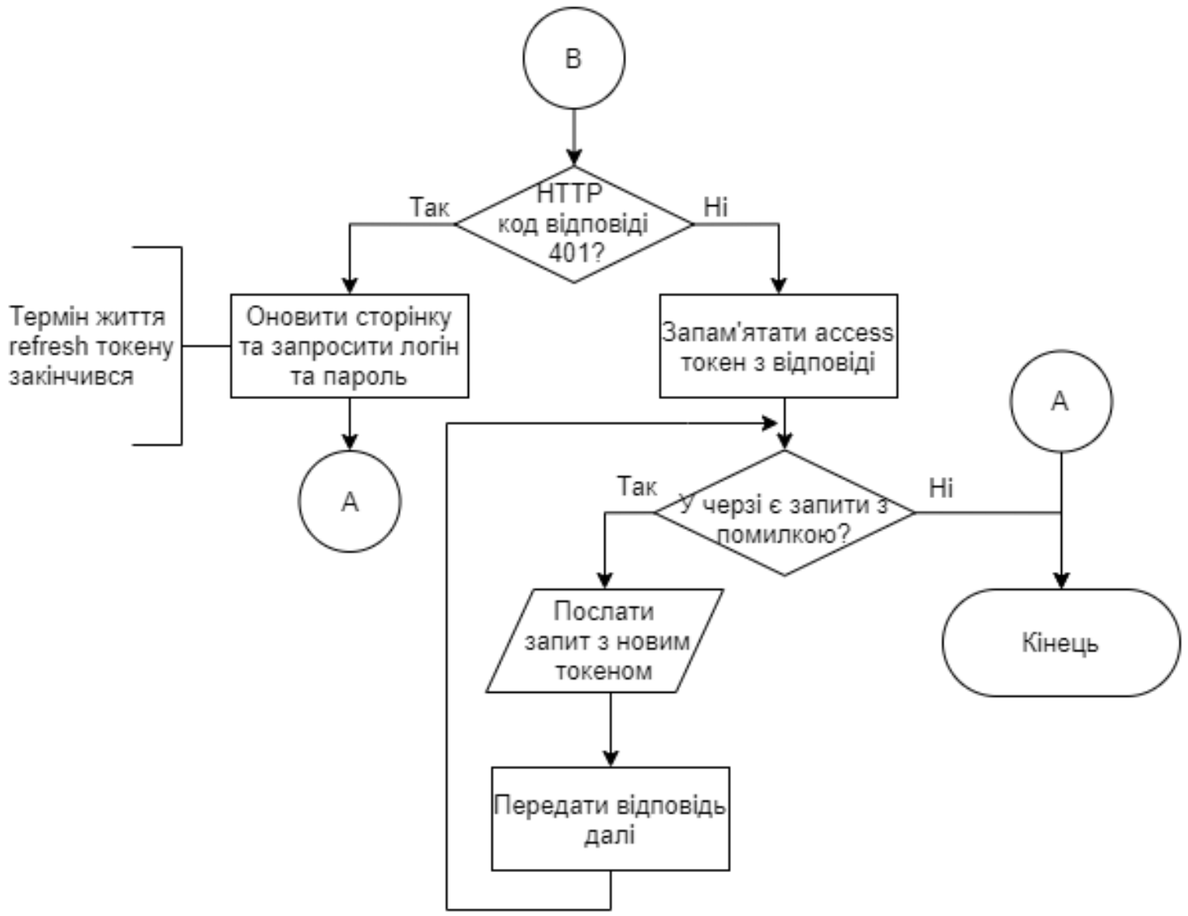
Блок-схема залежності об’єктів фронтенда

ІАЛЦ.467100.004 Д1

Аркушів 1

Київ 2020





					ІАЛЦ.467100.004 Д1
Змн.	*Арк.*	*№ докум.*	*Підпис*	*Дата*	
Розроб.		*Печена М.В.*			*Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення*
Перевір.		*Порєв В.М.*			*Блок-схема залежності об'єктів фронтенда*
Н. Контр.		*Сімоненко В.П.*			
Затверд.					

Літ.			*Арк.*	*Аркушів*
			1	*1*

НТУУ «КПІ ім. Ігоря Сікорського», ФІОТ ІП-62

ДОДАТОК 2

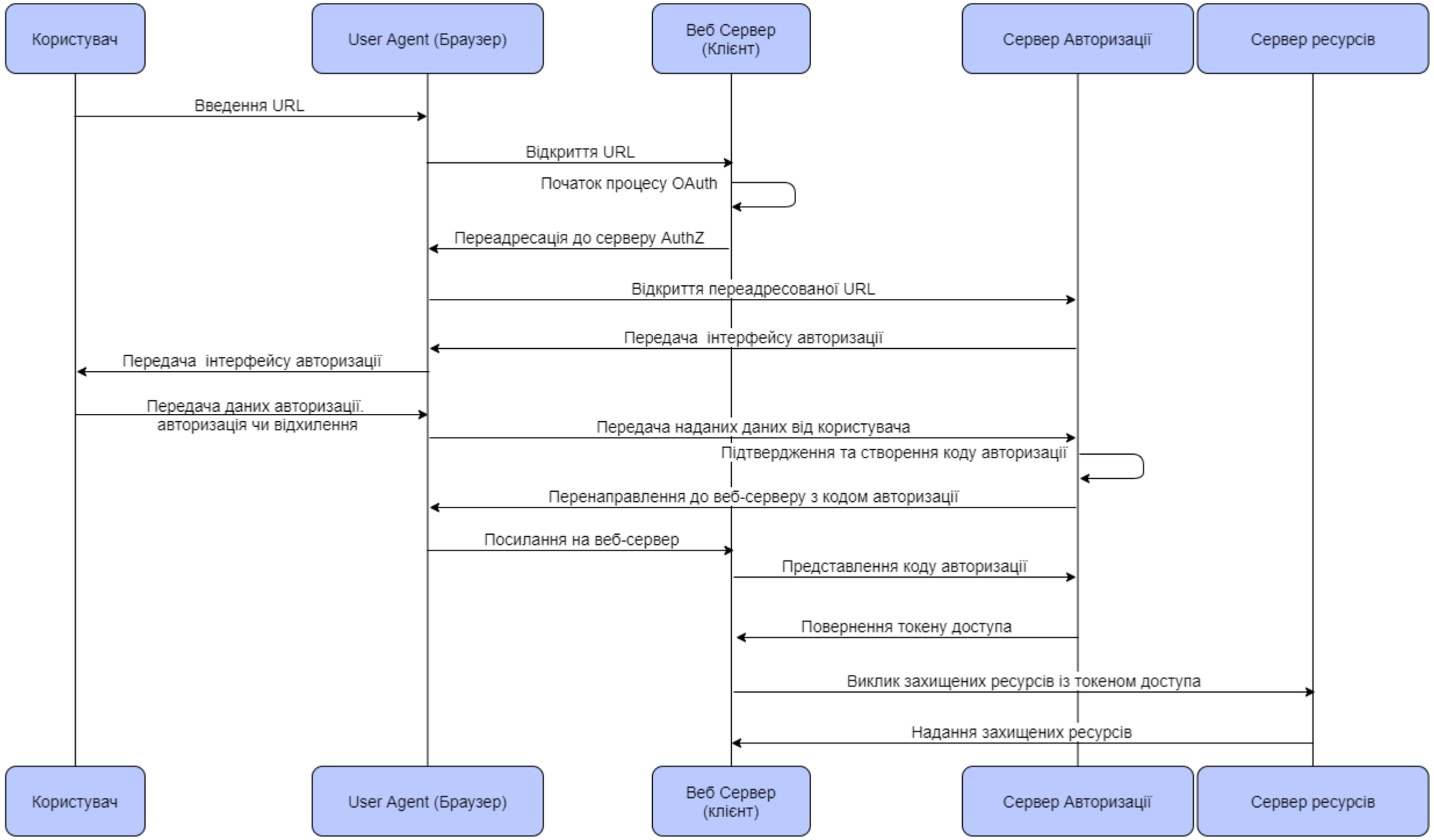
Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення

Ілюстрація взаємодії послідовності подій та їх обробка

ІАЛЦ.467100.005 Д2

Аркушів 1

Київ 2020



					ІАЛЦ.467100.005 Д2
Змн.	*Арк.*	*№ докум.*	*Підпис*	*Дата*	
Розроб.		*Печена М.В.*			*Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення*
Перевір.		*Порєв В.М.*			*Ілюстрація взаємодії послідовності подій та їх обробка*
Н. Контр.		*Сімоненко В.П.*			
Затверд.					

Літ.	*Арк.*	*Аркушів*
	1	*1*

НТУУ «КПІ ім. Ігоря Сікорського», ФІОТ ІП-62

ДОДАТОК 3

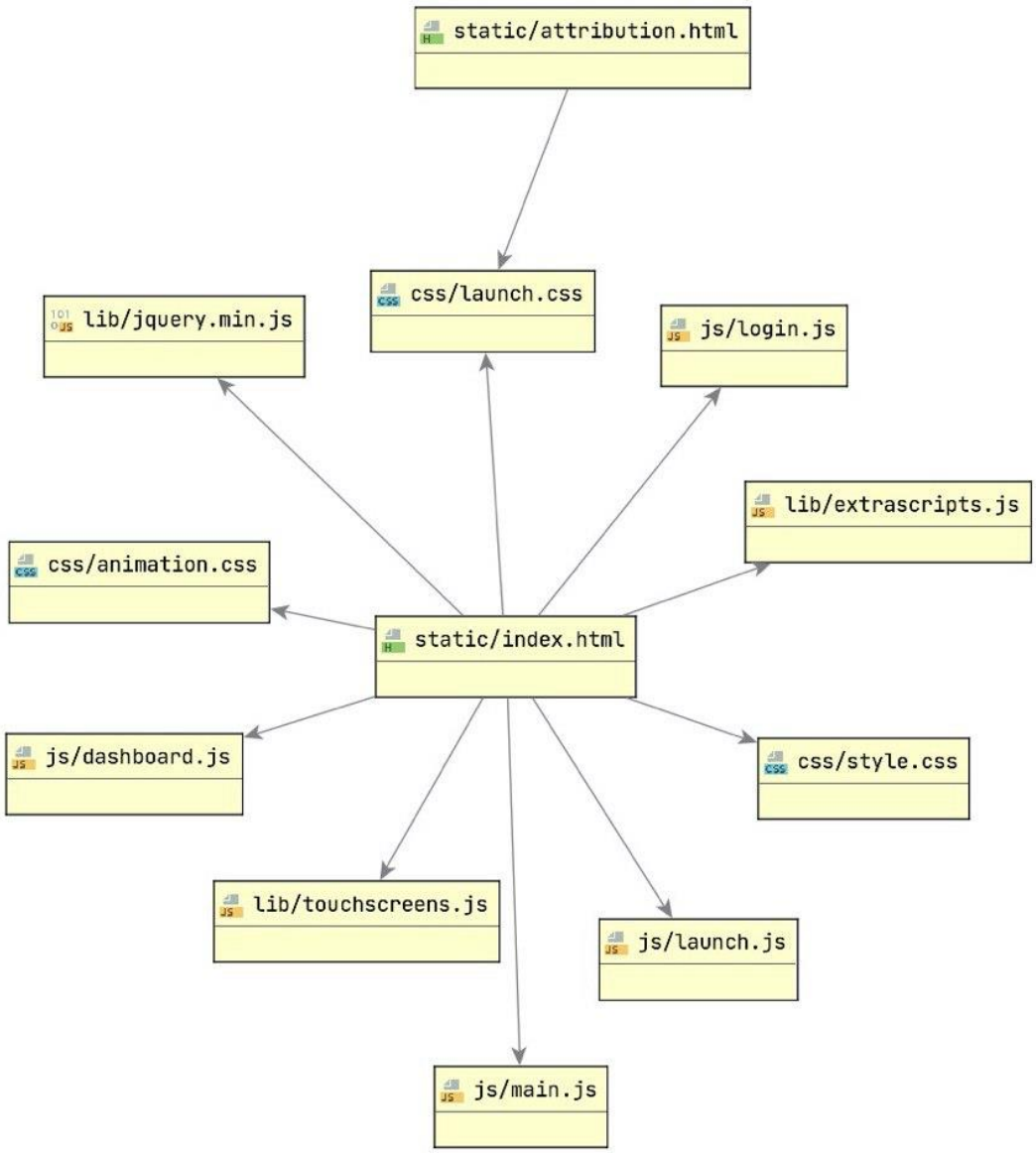
Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення

Схема взаємодії модулів проєкта

ІАЛЦ.467100.006 Д3

Аркушів 1

Київ 2020



					ІАЛЦ.467100.006 Д3	
Змн.	Арк.	№ докум.	Підпис	Дата		
Розроб.		Печена М.В.			Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення Схема взаємодії модулів проєкта	Літ. / Арк. 1 / Аркушів 1
Перевір.		Порєв В.М.				
Н. Контр.		Сімоненко В.П.				НТУУ «КПІ ім. Ігоря Сікорського», ФІОТ ІП-62
Затверд.						

ДОДАТОК 4

Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення

Лістинг

ІАЛЦ.467100.007 Д4

Аркушів 55

Київ 2020

index.html

```
<!doctype html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1"/>
        <title></title>

        <meta name="description" content=""/>
        <meta http-equiv="X-UA-Compatible" content="IE=edge"/>

        <link rel="stylesheet" href="css/jquery.bxslider.css" rel="stylesheet" />
        <!-- jQuery -->
        <script type="text/javascript" src="https://code.jquery.com/jquery-2.2.4.min.js"></script>
        <!-- BxSlider -->
        <script type="text/javascript" src="js/jquery.bxslider.min.js"></script>
        <script type="text/javascript" src="js/jquery.easydropdown.min.js"></script>

        <link rel="stylesheet" type="text/css" href="css/style.css" />
        <link rel="stylesheet" type="text/css" href="css/easydropdown.css"/>
        <link rel="stylesheet" type="text/css" href="css/hotspot-map.min.css"/>
        <script src="js/main.js"></script>
    </head>
<body>
    <div class="new_nav">
        <div class="mobile-button">
            <img src="img/logo2.png"/>
            <div id="mobile-toggle">
                <span></span>
            </div>
        </div>
        <ul id="menu">
            <li><a data-scroll href="#home"><img src="img/logo2.png"/></a></li>
            <li><a data-scroll href="#design">Design</a></li>
            <li><a data-scroll href="#apartment">Apartments</a></li>
            <li><a data-scroll href="#location">Location</a></li>
            <li><a data-scroll href="#lifestyle">Lifestyle</a></li>
            <li><a data-scroll href="#team">The Team</a></li>
            <li><a data-scroll href="#enquire">Enquire Today</a></li>
        </ul>
    </div>
    <div id="home" class="main_block">
        <div class="slider" id="one">
            <ul class="bxslider">
              <li style="background-image: url('img/slide1_1.jpg');"></li>
              <li style="background-image: url('img/full2.jpg');"></li>
            </ul>
        </div>
        <div id="dark">
            <p class="topAddr">1562 - 1568 CANTERBURY ROAD PUNCHBOWL</p>
            <div id="logo"></div>
            <h4>TO FIND OUT MORE</h4>
            <a data-scroll href="#design" class="scroll-down">
                        <img src="img/down-arrow.png" width="84" alt="arrow"/>
                </a>
        </div>
    </div>
    <div class="clear"></div>
    <div id="menu2">
        <div id="inner_menu">
```

					ІАЛЦ.467100.007 Д4			
Змн.	*Арк.*	*№ докум.*	*Підпис*	*Дата*				
Розроб.		*Печена М.В.*			*Веб-сервіс із мікросервісною архітектурою для моделювання нейромереж широкого призначення Текст програми*	*Літ.*	*Арк.*	*Аркушів*
Перевір.		*Порєв В.М.*					*1*	*55*
						НТУУ «КПІ ім. Ігоря Сікорського», ФІОТ ІП-62		
Н. Контр.		*Сімоненко В.П.*						
Затверд.								

```
            <li><a data-scroll href="#design">Design</a></li>
            <li><a data-scroll href="#apartment">Apartments</a></li>
            <li><a data-scroll href="#location">Location</a></li>
            <li><a data-scroll href="#lifestyle">Lifestyle</a></li>
            <li><a data-scroll href="#team">The Team</a></li>
            <li><a data-scroll href="#enquire">Enquire Today</a></li>
        </div>
    </div>
    <div class="clear"></div>
    <div id="design" class="main_block">
        <div class="center">
            <h2 class="caption">Design excellence &#38; affordability<br />
            have a brand new address</h2>
            <div class="slider" id="two">
                <ul class="bxslider">
                  <li style="background-image: url('img/slide1_1.jpg');"></li>
                  <li style="background-image: url('img/slide1_2.jpg');"></li>
                  <li style="background-image: url('img/slide1_3.jpg');"></li>
                </ul>
            </div>
            <h2>Setting a new benchmark in Affordable Value, Quality &#38; Style</h2>
            <p>Experience a bold new approach to affordable urban living at the brand new ASTORIA
Punchbowl. Designed by cutting edge architects Fox Johnston, these exquisite 1 and 2 bedroom
apartments focus on maximum use of space and flexibility at an affordable price. Designed with
practicality in mind, all apartments feature versatile enclosed loggias, which capitalise on
effective solar, acoustic and ventilation control for every resident. The rooftop features a herb
and vegetable garden, screened natural drying area and barbecue facilities all with district
views.<br /><br />
            Declared to be a landmark development for Punchbowl,  Its design will set a new
benchmark for residential construction and design, as it connects effortlessly with its surrounds,
all only a short stroll to trains, shops, cosmopolitan eateries and leafy riverside reserves. Also
most conveniently minutes to the M5.</p>
        </div>
    </div>
    <div class="clear"></div>
    <div id="apartment" class="main_block">
        <div class="center">
            <h2 class="caption2">Intelligent indoor and outdoor layouts Loggias create<br />
             seamless living efficiency and space with adaptability...<br />
            ‘Architectural Standout  Features’</h2>
            <div class="slider" id="three">
                <ul class="bxslider">
                  <li style="background-image: url('img/slide2_1.jpg');"></li>
                  <li style="background-image: url('img/slide2_2.jpg');"></li>
                </ul>
            </div>
            <h2 class="caption3">Light, Air, Innovation... Exquisite  Yet Affordable</h2>
            <p>A lifestyle of absolute comfort and flexibility is found in the bright, stylish
interiors of ASTORIA Punchbowl. Each generous living area is lined with glass panels that flow into
a loggia/dining area. Inspired architectural consideration has been given to the loggias to maximise
light and amenity while providing perfect ventilation and acoustic control, as you desire. </p>
        </div>
        <img id="full" src="img/full3.jpg" alt="full"/>
        <div class="center">
            <h2 class="caption4">Effortless living &#38; entertaining<br />
                Quality Design Functional Use of Space</h2>
            <div class="slider" id="four">
                <ul class="bxslider">
                  <li style="background-image: url('img/slide3_1.jpg');"></li>
                  <li style="background-image: url('img/slide3_2.jpg');"></li>
                </ul>
            </div>
            <h2 class="caption5">Light, Space &#38; Flexibility</h2>
            <p class="sentence">Streamlined kitchens are equipped for delicious cooking with gas
cooktops and oven. Sleek bathrooms serve spacious bedrooms with built-in robes, and neat study nooks
```

```
can be used for a variety of purposes. A soothing antidote to the stresses of the outside world, the
interiors are finished with cleverly concealed air conditioning units, main flooring is a classic,
stunning and natural timber.</p>
        </div>
    </div>
    <div id="location" class="main_block">
        <div class="center">
            <div class="block">
                <h2>Moments to Roselands or Bankstown</h2>
                <p>You'll be spoiled for choice once you've moved into your new home at ASTORIA
Punchbowl. The busy Punchbowl retail hub at the junction of Punchbowl Road and The Boulevarde boasts
plenty of shopping and dining options.  Supermarkets have all essentials and the Punchbowl Fruit
Palace is filled with fresh produce. Home to a number of thriving communities, Punchbowl has a range
of delights to discover and explore. If you like exotic food, the Middle Eastern cuisine of
Punchbowl is famed throughout Sydney. Lebanese delicacies are available for you to try at eateries
including El Jannanah Restaurant or satisfy your sweet tooth at the famous Rableh Sweet Shop. If you
need to do a more comprehensive shop, Roselands or Bankstown's Central Shopping Centre is a five
minute drive away with a range of major brand boutiques, department stores and supermarkets. </p>
            </div>
            <div class="block2">
                <img class="img1" src="img/slide4_1.jpg" alt="slide"/>
                <img class="img2" src="img/slide4_2.jpg" alt="slide"/>
            </div>
            <img class="img3" src="img/slide4_3.jpg" alt="slide"/>
            <div class="clear"></div>
            <img class="img4" src="img/slide4_4.jpg" alt="slide"/>
            <img class="img5" src="img/slide4_5.jpg" alt="slide"/>
        </div>
    </div>
    <div class="clear"></div>
    <div id="map" class="main_block">
          <div class="hs-wrap hs-loading" id="hotspot-48">
                <img src="img/map.jpg"/>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="25" data-y="67">
                    The Croatian Club
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="26.5" data-y="53">
                    Punchbowl Park
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="41" data-y="52">
                    Zoom Carwash Cafe
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="36" data-y="33">
                    Punchbowl Public School
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="13" data-y="27">
                    Punchbowl Station
                </div>
                <!------------------------5------------------------->
                <div class="hs-spot-object" data-height="30" data-popup-position=
```

```
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="11" data-y="30">
                    Lahori Dhaba
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="15" data-y="26">
                    Rashays Punchbowl
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="17" data-y="26">
                    Woolworths Punchbowl
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="18" data-y="22">
                    Wiley Park Girls High School
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="40" data-y="19">
                    Wiley Park
                </div>
                <!------------------------10------------------------->
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="62" data-y="21">
                    Roselands Shopping Centre
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "left" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="60" data-y="19">
                    Roselands Aquatic Centre
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "left" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="33.5" data-y="16">
                    Canterbury Hospital
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "left" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="56" data-y="11">
                    Sydney Airport
                </div>
                <div class="hs-spot-object" data-height="30" data-popup-position=
                "right" data-tint-color="#d6ede6" data-tooltip-auto-width="true"
                data-tooltip-width="200" data-type="spot" data-visible="visible"
                data-width="30" data-x="13" data-y="10">
                    Sydney CBD
                </div>
            </div>
        </div>
        <div class="clear"></div>
        <div id="lifestyle" class="main_block">
            <div class="center">
```

```
            <img class="img1" src="img/slide5_1.jpg" alt="slide"/>
            <div class="block">
                <h2>Central to major transport routes</h2>
                <p>Only 17 Kilometres South West of the Sydney CBD, ASTORIA Punchbowl has everything
you need for daily commuting, shopping and schooling. Getting around the area is exceptionally
straightforward. Three bus routes are within a stroll and city trains leave every 12 minutes from
Punchbowl Station at peak hour. Four minutes away, the M5 Motorway provides rapid access for drivers
to Hurstville, Sydney Airport and the Sydney CBD.<br /><br />
                The main retail strip also has an exciting selection of eateries and grocers
specialising in Middle Eastern and Asian cuisine. Savour Vietnamese pho at Pho Pasteur on Chapel
Road South, or enjoy Sunday yum cha at Great Century Restaurant on Greenfield Parade.</p>
                <div class="clear"></div>
            </div>
            <div class="clear"></div>
            <div class="block2">
                <div class="block3">
                    <img class="img2" src="img/slide5_2.jpg" alt="slide"/>
                    <img class="img3" src="img/slide5_3.jpg" alt="slide"/>
                    <img class="img4" src="img/slide5_4.jpg" alt="slide"/>
                </div>
                <img class="img5" src="img/slide5_5.jpg" alt="slide"/>
            </div>
        </div>
    </div>
    <div class="clear"></div>
    <div id="team" class="main_block">
        <div class="center">
            <h2 class="caption3">A progressive vision for the future</h2>
            <div class="half spec">
                <img src="img/team.jpg" alt="Architect"/>
                <h2>Architect</h2>
                <p>Fox Johnston is a young Sydney-based architecture &#38; design practice that
combines an intelligent, pragmatic response to each client’s brief with a thorough understanding of
the opportunities created by site and context. We work across a range of projects, from large
private houses to adaptive re-use of heritage buildings to major multiple residence projects. Our
work is driven a passion for light, space, natural materials and finely crafted living spaces,
coupled with an acute awareness of the sustainability challenges facing the built environment today.
The practice specialises in creating well-resolved, light-filled, tactile and thought-provoking
buildings – both single houses and multi-residential projects - for people to live work & play in.
Living in a Fox Johnston-designed building is the perfect antidote to the stresses of today’s
complex &#38; challenging world.</p>
            </div>
            <div class="half spec">
                <img src="img/team2.jpg" alt="Developer"/>
                <h2>Developer</h2>
                <p>Piper Property Group is a leading boutique property development management
company. Established in 1996 they have been delivering successful developments with more than 25
years experience. They are a dynamic and progressive organisation who have become a significant
participant in Australia’s major property markets by working closely with clients, investors and
communities. Our progressive team of professionals and experience has established exclusive
relationships ensuring “on the ground” services which provide a truly collaborative approach and
project delivery over the test of time.</p>
            </div>
            <div class="clear"></div>
            <div class="half">
                <img class="partner" src="img/partner.png" alt="partner"/>
            </div>
            <div class="half">
                <img class="partner" src="img/partner2.png" alt="partner"/>
            </div>
        </div>
    </div>
    <div class="clear"></div>
    <div id="enquire" class="main_block">
        <div class="center">
```

```
            <h2 class="caption4">For further information about this execiting release
            please fill the below form or contact us for your exclusive introduction </h2>
            <div class="clear"></div>
            <form  id="contactform" action="" method="post">
                        <span class="inputwrap">
                              <input class="vfield" type="text" placeholder="First Name*"
name="fname" required/>
                        </span>
                        <span class="inputwrap">
                              <input class="vfield" type="text" placeholder="Last Name*"
name="lname" required/>
                        </span>
                        <span class="inputwrap">
                              <input class="vfield" type="text" placeholder="Email*" name="email"
required/>
                        </span>
                        <span class="inputwrap">
                              <input class="vfield" type="text" placeholder="Mobile*" name="phone"
required/>
                        </span>
                        <span class="inputwrap">
                              <input class="vfield" type="text" placeholder="Current Post Code*"
name="postcode" required/>
                        </span>
                  <span class="inputwrap">
                              <select name="whoami" class="dropdown" size="1">
                              <option value="">I am*</option>
                              <option value="First home buyer">First home buyer</option>
                              <option value="Investor">Investor</option>
                              <option value="Owner Occupier">Owner Occupier</option>
                        </select>
                  </span>
                  <span class="inputwrap">
                              <select name="interested" class="dropdown" size="1">
                              <option value="">Interested in ?*</option>
                              <option value="1 Bedroom">1 Bedroom</option>
                              <option value="2 Bedroom">2 Bedroom</option>
                        </select>
                  </span>
                  <span class="inputwrap">
                              <select name="budget" class="dropdown" size="1">
                              <option value="">My Budget Range*</option>
                              <option value="$370,000 - $470,000">$370,000 - $470,000</option>
                              <option value="$471,000 - $585,000">$471,000 - $585,000</option>
                              <option value="$590,000 - $650,000">$590,000 - $650,000</option>
                        </select>
                  </span>
                  <span class="inputwrap">
                              <select name="how_hear" class="dropdown" size="1">
                              <option value="">How did you hear about us*</option>
                              <option value="Domain.com.au">Domain.com.au</option>
                              <option value="Signage">Signage</option>
                              <option value="Direct mail">Direct mail</option>
                              <option value="Referral">Referral</option>
                              <option value="Sydney Morning Herald">Sydney Morning Herald</option>
                                    <option value="Daily Telegraph">Daily Telegraph</option>
                                    <option value="Ethnic Publication">Ethnic Publication</option>
                        </select>
                  </span>
                        <button id="register-now-button" type="submit">SUBMIT</button>
                        <div id="preloader" style="display:none; text-align:center;"><img
src="img/loader.gif" /></div>
                  </form>
            <div class="block">
                  <h2 class="caption5">Enquiries</h2>
```

```
                <p class="tel">1300 010 888</p>
                <div class="clear"></div>
                <h2 class="caption6">Sales &#38; Display</h2>
                <div class="clear"></div>
                <p class="visit">VISIT OUR DISPLAY SUITE<br /><br />
                Open Thursday - Sunday,<br />
                11.00am to 3.00pm<br /><br />
                11 Weyland Street,<br />
                Punchbowl NSW 2196</p>
                <div class="clear"></div>
                <img class="partner" src="img/partner3.png" alt="partner"/>
            </div>
        </div>
    </div>
    <div class="clear"></div>
    <footer>
        <div class="center">
            <p class="notice">Privacy   Disclaimer</p>
            <p class="notice2">creative by kokky</p>
            <img src="img/logo4.png" alt="logo"/>
        </div>
    </footer>
    <script type="text/javascript" src="js/hotspot-map.min.js"></script>
     <!-- Smooth Scroll -->
    <script src='js/smooth-scroll.js'></script>
    <script type="text/javascript" src="js/main.js"></script>
</body>
</html>
```

Main.js

```
$(document).ready(function(){
    /*-------------Map---------------*/
    $("#hotspot-48").hotspot({ "show_on" : "mouseover", "responsive" : true });
    /*-----------------Menu------------------------*/
    $(function() {
        if (Number($(window).width()) <= 700){
            $("#menu").css("margin-top", "0");
            $("#mobile-toggle").click(function(){
                $(".new_nav").toggleClass("show");
            });
            //Сворачиваем меню при нажатии
            $("#menu li a").on("click", function(){
                $(".new_nav").toggleClass("show");
            });
            //Появление меню
            $(window).scroll(function() {
                var scrollTop = $(window).scrollTop();
            });
        } else{
            $(window).scroll(function() {
                var blocks = ["design", "apartment", "location", "lifestyle", "team", "enquire"];
                var scrollTop = $(window).scrollTop();
                //Появление меню
                if (scrollTop <= $("#design").offset().top){
                    $("#menu").css("margin-top", "-64px");
                } else $("#menu").css("margin-top", "0");

                for(var i = 0; i < blocks.length; i++){
                    var top = $("#"+blocks[i]).offset().top;
                    if(scrollTop > top){
                        $("#menu a").css("color", "#000");
                        $("#menu a[href='#"+blocks[i]+"']").css("color", "#d3ac5b");
```

```
                    } else if((i == 0) && (scrollTop < top)){
                        $("#menu a").css("color", "#000");
                    }
                }
            });
        }
    });
    /*Slider*/
    $('.bxslider').bxSlider({
         mode: 'fade',
         pager: false,
         controls: false,
         auto: true,
         speed: 1000
   });
    /*-----------------Scroll-----------------*/
    var ofset = 0;
    if (Number($(window).width()) <= 700) ofset = 30;
    smoothScroll.init({
        selector: '[data-scroll]',
         speed: 1000,
         easing: 'easeInOutCubic',
         offset: ofset,
         updateURL: true,
         callbackBefore: function ( toggle, anchor ) {},
         callbackAfter: function ( toggle, anchor ) {}
 });
});
```

hotspot-map.min.js

```
(function(d)
{function e()
 {this.isField=!0;
 this.keyboardMode=this.hasLabel=this.cutOff=this.disabled=this.inFocus=this.down=!1;
 this.nativeTouch=!0;
 this.wrapperClass="dropdown";
 this.onChange=null}e.prototype={constructor:e,instances:{},init:function(a,c){var
b=this;d.extend(b,c);
 b.$select=d(a);b.id=a.id;b.options=[];
 b.$options=b.$select.find("option");
 b.isTouch="ontouchend"in document;
 b.$select.removeClass(b.wrapperClass+" dropdown");
 b.$select.is(":disabled")&&(b.disabled=!0);
 b.$options.length&&(b.$options.each(function(a){var c=d(this);
 c.is(":selected")&&(b.selected={index:a,title:c.text()},b.focusIndex=a);
 c.hasClass("label")&&0==a?(b.hasLabel=!0,b.label=c.text(),c.attr("value","")):
        b.options.push({domNode:c[0],title:c.text(),value:c.val(),
        selected:c.is(":selected")
 })
}),
b.selected||(b.selected={index:0,title:b.$options.eq(0).text()},b.focusIndex=0),
b.render())},
render:function()
{
 var a=this;
 a.$container=a.$select.wrap('<div class="'+a.wrapperClass+(a.isTouch&&a.nativeTouch?" touch":"")
        +(a.disabled?" disabled":"")+'"><span class="old"/></div>').parent().parent();
 a.$active=d('<span class="selected">'+a.selected.title+"</span>")
        .appendTo(a.$container);
 a.$carat=d('<span class="carat"/>')
        .appendTo(a.$container);
 a.$scrollWrapper=d("<div><ul/></div>")
        .appendTo(a.$container);
        a.$dropDown=a.$scrollWrapper.find("ul");
```

```
        a.$form=a.$container.closest("form");
        d.each(a.options,function(){a.$dropDown.append("<li"+(this.selected?'
class="active"':"")+">"+this.title+"</li>")});
        a.$items=a.$dropDown.find("li");
        a.cutOff&&a.$items.length>a.cutOff&&a.$container.addClass("scrollable");
 a.getMaxHeight();
 a.isTouch&&a.nativeTouch?a.bindTouchHandlers():a.bindHandlers()
},
getMaxHeight:
 function()
 {
        for(i=this.maxHeight=0;i<this.$items.length;i++)
        {
               var a=this.$items.eq(i);
               this.maxHeight+=a.outerHeight();
               if(this.cutOff==i+1)break
        }
 }
,bindTouchHandlers:function()
{
 var a=this;a.$container.on("click.easyDropDown",function()
 {
        a.$select.focus()});
        a.$select.on({change:function()
        {
               var c=d(this).find("option:selected"),b=c.text(),c=c.val();
               a.$active.text(b);
               "function"===typeof a.onChange&&a.onChange
                      .call(a.$select[0],{title:b,value:c})
        },focus:function()
        {
               a.$container.addClass("focus")
        },blur:function()
        {
               a.$container.removeClass("focus")
        }
 })
},bindHandlers:function()
{
 var a=this;
 a.query="";
 a.$container.on({"click.easyDropDown":function()
 {
        a.down||a.disabled?a.close():a.open()
 },"mousemove.easyDropDown":function(){a.keyboardMode&&(a.keyboardMode=!1)}
});
d("body").on("click.easyDropDown."+a.id,function(c){c=d(c.target);
var b=a.wrapperClass.split(" ").join(".");
!c.closest("."+b).length&&a.down&&a.close()});
a.$items.on({"click.easyDropDown":function()
{
 var c=d(this).index();
 a.select(c);
 a.$select.focus()
},"mouseover.easyDropDown":function()
{
 if(!a.keyboardMode)
 {
        var c=d(this);
        c.addClass("focus").siblings().removeClass("focus");
        a.focusIndex=c.index()
 }
},"mouseout.easyDropDown":function()
{
 a.keyboardMode||d(this).removeClass("focus")}});
```

```
}
disable:function()
{
 this.disabled=!0;
 this.$container.addClass("disabled");
 this.$select.attr("disabled",!0);
 this.down||this.close()
},enable:function()
{
 this.disabled=!1;
 this.$container.removeClass("disabled");
 this.$select.attr("disabled",!1)}
};
var f=function(a,c)
{
 a.id=a.id?a.id:"EasyDropDown"+("00000"+(16777216*Math.random()<<0).toString(16)).substr(-
6).toUpperCase();
 var b=new e;b.instances[a.id]||(b.instances[a.id]=b,b.init(a,c))};
 d.fn.easyDropDown=function()
 {
        var a=arguments,c=[],b;b=this.each(function(){if(a&&"string"===typeof a[0])
        {
               var b=e.prototype.instances[this.id][a[0]](a[1],a[2]);b&&c.push(b)
        }
        else f(this,a[0])});
        return c.length?1<c.length?c:c[0]:b
 };
 d(function()
 {
        "function"!==typeof
Object.getPrototypeOf&&(Object.getPrototypeOf="object"===typeof"test".__proto__?function(a)
        {
               return a.__proto__
        }:function(a)
        {
               return a.constructor.prototype
        });
        d("select.dropdown").each(function(){var a=d(this).attr("data-
settings");settings=a?d.parseJSON(a):{};f(this,settings)})})})(jQuery);
```

Smooth-scroll.js

```
(function (root, factory) {
 if ( typeof define === 'function' && define.amd ) {
        define('smoothScroll', factory(root));
 } else if ( typeof exports === 'object' ) {
        module.exports = factory(root);
 } else {
        root.smoothScroll = factory(root);
 }
})(window || this, function (root) {

 'use strict';

 //
 // Variables
 //

 var smoothScroll = {}; // Object for public APIs
 var supports = !!document.querySelector && !!root.addEventListener; // Feature test
 var settings, eventTimeout, fixedHeader;

 // Default settings
 var defaults = {
```

```
        speed: 500,
        easing: 'easeInOutCubic',
        offset: 0,
        updateURL: true,
        callbackBefore: function () {},
        callbackAfter: function () {}
};


//
// Methods
//

/**
 * A simple forEach() implementation for Arrays, Objects and NodeLists
 * @private
 * @param {Array|Object|NodeList} collection Collection of items to iterate
 * @param {Function} callback Callback function for each iteration
 * @param {Array|Object|NodeList} scope Object/NodeList/Array that forEach is iterating over (aka
`this`)
 */
var forEach = function (collection, callback, scope) {
        if (Object.prototype.toString.call(collection) === '[object Object]') {
                for (var prop in collection) {
                        if (Object.prototype.hasOwnProperty.call(collection, prop)) {
                                callback.call(scope, collection[prop], prop, collection);
                        }
                }
        } else {
                for (var i = 0, len = collection.length; i < len; i++) {
                        callback.call(scope, collection[i], i, collection);
                }
        }
};

/**
 * Merge defaults with user options
 * @private
 * @param {Object} defaults Default settings
 * @param {Object} options User options
 * @returns {Object} Merged values of defaults and options
 */
var extend = function ( defaults, options ) {
        var extended = {};
        forEach(defaults, function (value, prop) {
                extended[prop] = defaults[prop];
        });
        forEach(options, function (value, prop) {
                extended[prop] = options[prop];
        });
        return extended;
};

/**
 * Get the closest matching element up the DOM tree
 * @param {Element} elem Starting element
 * @param {String} selector Selector to match against (class, ID, or data attribute)
 * @return {Boolean|Element} Returns false if not match found
 */
var getClosest = function (elem, selector) {
        var firstChar = selector.charAt(0);
        for ( ; elem && elem !== document; elem = elem.parentNode ) {
                if ( firstChar === '.' ) {
                        if ( elem.classList.contains( selector.substr(1) ) ) {
                                return elem;
```

```
                }
            } else if ( firstChar === '#' ) {
                if ( elem.id === selector.substr(1) ) {
                    return elem;
                }
            } else if ( firstChar === '[' ) {
                if ( elem.hasAttribute( selector.substr(1, selector.length - 2) ) ) {
                    return elem;
                }
            }
        }
        return false;
};

/**
 * Get the height of an element
 * @private
 * @param  {Node]} elem The element
 * @return {Number}     The element's height
 */
var getHeight = function (elem) {
        return Math.max( elem.scrollHeight, elem.offsetHeight, elem.clientHeight );
};

/**
 * Escape special characters for use with querySelector
 * @private
 * @param {String} id The anchor ID to escape
 * @author Mathias Bynens
 * @link https://github.com/mathiasbynens/CSS.escape
 */
var escapeCharacters = function ( id ) {
        var string = String(id);
        var length = string.length;
        var index = -1;
        var codeUnit;
        var result = '';
        var firstCodeUnit = string.charCodeAt(0);
        while (++index < length) {
            codeUnit = string.charCodeAt(index);
            // Note: there’s no need to special-case astral symbols, surrogate
            // pairs, or lone surrogates.

            // If the character is NULL (U+0000), then throw an
            // `InvalidCharacterError` exception and terminate these steps.
            if (codeUnit === 0x0000) {
                throw new InvalidCharacterError(
                    'Invalid character: the input contains U+0000.'
                );
            }

            if (
                // If the character is in the range [\1-\1F] (U+0001 to U+001F) or is
                // U+007F, […]
                (codeUnit >= 0x0001 && codeUnit <= 0x001F) || codeUnit == 0x007F ||
                // If the character is the first character and is in the range [0-9]
                // (U+0030 to U+0039), […]
                (index === 0 && codeUnit >= 0x0030 && codeUnit <= 0x0039) ||
                // If the character is the second character and is in the range [0-9]
                // (U+0030 to U+0039) and the first character is a `-` (U+002D), […]
                (
                    index === 1 &&
                    codeUnit >= 0x0030 && codeUnit <= 0x0039 &&
                    firstCodeUnit === 0x002D
                )
```

```
                ) {
                        // http://dev.w3.org/csswg/cssom/#escape-a-character-as-code-point
                        result += '\\' + codeUnit.toString(16) + ' ';
                        continue;
                }

                // If the character is not handled by one of the above rules and is
                // greater than or equal to U+0080, is `-` (U+002D) or `_` (U+005F), or
                // is in one of the ranges [0-9] (U+0030 to U+0039), [A-Z] (U+0041 to
                // U+005A), or [a-z] (U+0061 to U+007A), […]
                if (
                        codeUnit >= 0x0080 ||
                        codeUnit === 0x002D ||
                        codeUnit === 0x005F ||
                        codeUnit >= 0x0030 && codeUnit <= 0x0039 ||
                        codeUnit >= 0x0041 && codeUnit <= 0x005A ||
                        codeUnit >= 0x0061 && codeUnit <= 0x007A
                ) {
                        // the character itself
                        result += string.charAt(index);
                        continue;
                }

                // Otherwise, the escaped character.
                // http://dev.w3.org/csswg/cssom/#escape-a-character
                result += '\\' + string.charAt(index);

        }
        return result;
 };

 /**
  * Calculate the easing pattern
  * @private
  * @link https://gist.github.com/gre/1650294
  * @param {String} type Easing pattern
  * @param {Number} time Time animation should take to complete
  * @returns {Number}
  */
 var easingPattern = function ( type, time ) {
        var pattern;
        if ( type === 'easeInQuad' ) pattern = time * time; // accelerating from zero velocity
        if ( type === 'easeOutQuad' ) pattern = time * (2 - time); // decelerating to zero velocity
        if ( type === 'easeInOutQuad' ) pattern = time < 0.5 ? 2 * time * time : -1 + (4 - 2 *
time) * time; // acceleration until halfway, then deceleration
        if ( type === 'easeInCubic' ) pattern = time * time * time; // accelerating from zero
velocity
        if ( type === 'easeOutCubic' ) pattern = (--time) * time * time + 1; // decelerating to
zero velocity
        if ( type === 'easeInOutCubic' ) pattern = time < 0.5 ? 4 * time * time * time : (time - 1)
* (2 * time - 2) * (2 * time - 2) + 1; // acceleration until halfway, then deceleration
        if ( type === 'easeInQuart' ) pattern = time * time * time * time; // accelerating from
zero velocity
        if ( type === 'easeOutQuart' ) pattern = 1 - (--time) * time * time * time; // decelerating
to zero velocity
        if ( type === 'easeInOutQuart' ) pattern = time < 0.5 ? 8 * time * time * time * time : 1 -
8 * (--time) * time * time * time; // acceleration until halfway, then deceleration
        if ( type === 'easeInQuint' ) pattern = time * time * time * time * time; // accelerating
from zero velocity
        if ( type === 'easeOutQuint' ) pattern = 1 + (--time) * time * time * time * time; //
decelerating to zero velocity
        if ( type === 'easeInOutQuint' ) pattern = time < 0.5 ? 16 * time * time * time * time *
time : 1 + 16 * (--time) * time * time * time * time; // acceleration until halfway, then
deceleration
        return pattern || time; // no easing, no acceleration
```

```
};

/**
 * Calculate how far to scroll
 * @private
 * @param {Element} anchor The anchor element to scroll to
 * @param {Number} headerHeight Height of a fixed header, if any
 * @param {Number} offset Number of pixels by which to offset scroll
 * @returns {Number}
 */
var getEndLocation = function ( anchor, headerHeight, offset ) {
	var location = 0;
	if (anchor.offsetParent) {
		do {
			location += anchor.offsetTop;
			anchor = anchor.offsetParent;
		} while (anchor);
	}
	location = location - headerHeight - offset;
	return location >= 0 ? location : 0;
};

/**
 * Determine the document's height
 * @private
 * @returns {Number}
 */
var getDocumentHeight = function () {
	return Math.max(
		document.body.scrollHeight, document.documentElement.scrollHeight,
		document.body.offsetHeight, document.documentElement.offsetHeight,
		document.body.clientHeight, document.documentElement.clientHeight
	);
};

/**
 * Convert data-options attribute into an object of key/value pairs
 * @private
 * @param {String} options Link-specific options as a data attribute string
 * @returns {Object}
 */
var getDataOptions = function ( options ) {
	return !options || !(typeof JSON === 'object' && typeof JSON.parse === 'function') ? {} :
JSON.parse( options );
};

/**
 * Update the URL
 * @private
 * @param {Element} anchor The element to scroll to
 * @param {Boolean} url Whether or not to update the URL history
 */
var updateUrl = function ( anchor, url ) {
	if ( history.pushState && (url || url === 'true') ) {
		history.pushState( null, null, [root.location.protocol, '//', root.location.host,
root.location.pathname, root.location.search, anchor].join('') );
	}
};

/**
 * Start/stop the scrolling animation
 * @public
 * @param {Element} toggle The element that toggled the scroll event
 * @param {Element} anchor The element to scroll to
 * @param {Object} options
```

```
  */
 smoothScroll.animateScroll = function ( toggle, anchor, options ) {

        // Options and overrides
        var settings = extend( settings || defaults, options || {} );  // Merge user options with
defaults
        var overrides = getDataOptions( toggle ? toggle.getAttribute('data-options') : null );
        settings = extend( settings, overrides );
        anchor = '#' + escapeCharacters(anchor.substr(1)); // Escape special characters and leading
numbers

        // Selectors and variables
        var anchorElem = anchor === '#' ? document.documentElement :
document.querySelector(anchor);
        var startLocation = root.pageYOffset; // Current location on the page
        if ( !fixedHeader ) { fixedHeader = document.querySelector('[data-scroll-header]'); }  //
Get the fixed header if not already set
        var headerHeight = fixedHeader === null ? 0 : ( getHeight( fixedHeader ) +
fixedHeader.offsetTop ); // Get the height of a fixed header if one exists
        var endLocation = getEndLocation( anchorElem, headerHeight, parseInt(settings.offset, 10)
); // Scroll to location
        var animationInterval; // interval timer
        var distance = endLocation - startLocation; // distance to travel
        var documentHeight = getDocumentHeight();
        var timeLapsed = 0;
        var percentage, position;

        // Update URL
        updateUrl(anchor, settings.updateURL);

        /**
         * Stop the scroll animation when it reaches its target (or the bottom/top of page)
         * @private
         * @param {Number} position Current position on the page
         * @param {Number} endLocation Scroll to location
         * @param {Number} animationInterval How much to scroll on this loop
         */
        var stopAnimateScroll = function (position, endLocation, animationInterval) {
               var currentLocation = root.pageYOffset;
               if ( position == endLocation || currentLocation == endLocation || (
(root.innerHeight + currentLocation) >= documentHeight ) ) {
                      clearInterval(animationInterval);
                      anchorElem.focus();
                      settings.callbackAfter( toggle, anchor ); // Run callbacks after animation
complete
               }
        };

        /**
         * Loop scrolling animation
         * @private
         */
        var loopAnimateScroll = function () {
               timeLapsed += 16;
               percentage = ( timeLapsed / parseInt(settings.speed, 10) );
               percentage = ( percentage > 1 ) ? 1 : percentage;
               position = startLocation + ( distance * easingPattern(settings.easing, percentage)
);
               root.scrollTo( 0, Math.floor(position) );
               stopAnimateScroll(position, endLocation, animationInterval);
        };

        /**
         * Set interval timer
         * @private
```

```
         */
        var startAnimateScroll = function () {
                settings.callbackBefore( toggle, anchor ); // Run callbacks before animating scroll
                animationInterval = setInterval(loopAnimateScroll, 16);
        };

        /**
         * Reset position to fix weird iOS bug
         * @link https://github.com/cferdinandi/smooth-scroll/issues/45
         */
        if ( root.pageYOffset === 0 ) {
                root.scrollTo( 0, 0 );
        }

        // Start scrolling animation
        startAnimateScroll();

};

/**
 * If smooth scroll element clicked, animate scroll
 * @private
 */
var eventHandler = function (event) {
        var toggle = getClosest(event.target, '[data-scroll]');
        if ( toggle && toggle.tagName.toLowerCase() === 'a' ) {
                event.preventDefault(); // Prevent default click event
                smoothScroll.animateScroll( toggle, toggle.hash, settings); // Animate scroll
        }
};

/**
 * On window scroll and resize, only run events at a rate of 15fps for better performance
 * @private
 * @param  {Function} eventTimeout Timeout function
 * @param  {Object} settings
 */
var eventThrottler = function (event) {
        if ( !eventTimeout ) {
                eventTimeout = setTimeout(function() {
                        eventTimeout = null; // Reset timeout
                        headerHeight = fixedHeader === null ? 0 : ( getHeight( fixedHeader ) +
fixedHeader.offsetTop ); // Get the height of a fixed header if one exists
                }, 66);
        }
};

/**
 * Destroy the current initialization.
 * @public
 */
smoothScroll.destroy = function () {

        // If plugin isn't already initialized, stop
        if ( !settings ) return;

        // Remove event listeners
        document.removeEventListener( 'click', eventHandler, false );
        root.removeEventListener( 'resize', eventThrottler, false );

        // Reset varaibles
        settings = null;
        eventTimeout = null;
        fixedHeader = null;
};
```

```
  /**
   * Initialize Smooth Scroll
   * @public
   * @param {Object} options User settings
   */
  smoothScroll.init = function ( options ) {

          // feature test
          if ( !supports ) return;

          // Destroy any existing initializations
          smoothScroll.destroy();

          // Selectors and variables
          settings = extend( defaults, options || {} ); // Merge user options with defaults
          fixedHeader = document.querySelector('[data-scroll-header]'); // Get the fixed header

          // When a toggle is clicked, run the click handler
          document.addEventListener('click', eventHandler, false );
          if ( fixedHeader ) { root.addEventListener( 'resize', eventThrottler, false ); }

  };


  //
  // Public APIs
  //

  return smoothScroll;

});
```

Style.css

```
@font-face {
  font-family: 'ElegantIcons';
  src:url('../fonts/ElegantIcons.eot');
  src:url('../fonts/ElegantIcons.eot?#iefix') format('embedded-opentype'),
          url('../fonts/ElegantIcons.woff') format('woff'),
          url('../fonts/ElegantIcons.ttf') format('truetype'),
          url('../fonts/ElegantIcons.svg#ElegantIcons') format('svg');
  font-weight: normal;
  font-style: normal;
}
@font-face{
    font-family: 'GothamBook';
    src: url("../fonts/GothamBook.eot");
    src: url("../fonts/GothamBook.eot") format("embedded-opentype"),
         url("../fonts/GothamBook.woff") format("woff"),
         url("../fonts/GothamBook.ttf") format("truetype"),
         url('../fonts/GothamBook.svg') format('svg');
    font-weight: normal;
    font-style: normal;
}
@font-face{
    font-family: 'GothamMedium';
    src: url("../fonts/gotham-medium.eot");
    src: url("../fonts/gotham-medium.eot") format("embedded-opentype"),
         url("../fonts/gotham-medium.woff") format("woff"),
         url("../fonts/gotham-medium.ttf") format("truetype"),
         url('../fonts/gotham-medium.svg') format('svg');
    font-weight: normal;
    font-style: normal;
```

```
}
@font-face{
    font-family: 'Perpetua-Italic';
    src: url("../fonts/Perpetua-Italic.eot");
    src: url("../fonts/Perpetua-Italic.eot") format("embedded-opentype"),
         url("../fonts/Perpetua-Italic.woff") format("woff"),
         url("../fonts/Perpetua-Italic.ttf") format("truetype"),
         url('../fonts/Perpetua-Italic.ttf.svg') format('svg');
    font-weight: normal;
    font-style: normal;
}
body{
    margin: 0;
    padding: 0;
    background: #fcfffb;
    text-align: center;
    font-family: 'GothamBook';
}
ul, li{
    list-style: none;
    padding: 0;
    margin: 0;
}
a{
    text-decoration: none;
}
.clear{
    clear: both;
}
p{
    display: inline-block;
    font-size: 18px;
    color: #fcfffb;
    line-height: 120%;
    margin: 0;
}
h2 {
    font-family: 'Perpetua-Italic';
    font-weight: 200;
    font-size: 48px;
    color: #d3ac5b;
}
h4 {
    font-weight: 100;
    font-size: 18px;
    color: #fcfffb;
    text-transform: uppercase;
}
.new_nav {
    position: fixed;
    top: 0;
    background: #fcfffb;
    width: 100%;
    z-index: 9999;
    transition: all 1000ms;
    -moz-transition: all 1000ms;
    -ms-transition: all 1000ms;
    -webkit-transition: all 1000ms;
}
#menu {
    max-width: 1340px;
    margin: 0 auto;
    padding: 24px 0 22px;
    text-align: right;
    transition: all 1000ms;
```

```
    -moz-transition: all 1000ms;
    -ms-transition: all 1000ms;
    -webkit-transition: all 1000ms;
    margin-top: -64px;
}
#menu li, #menu2 li{
    display: inline-block;
    list-style: none;
    z-index: 999;
    text-transform: uppercase;
    letter-spacing: 0.01em;
    padding: 0 20px 0 0px;
}
#menu li:after, #menu2 li:after{
    content: "|";
    padding-left: 20px;
    color: #d3ac5b;
    font-weight: bold;
}
#menu li:first-child:after, #menu li:last-child:after, #menu2 li:last-child:after {
    content: "";
}
#menu li:first-child{
    float: left;
    margin-top: -22px;
}
#menu li:last-child{
    padding: 0;
}
#menu li a, #menu2 li a{
    font-family: 'GothamMedium';
    font-size: 14px;
    color: #000;
    transition: ease color .5s;
}
.mobile-button{
 display: none;
}
#mobile-toggle, .mobile-button img{
    cursor: pointer;
    font-size: 28px;
    position: absolute;
    right: 22px;
    top: 11px;
    z-index: 99999;
}
.mobile-button img{
    cursor: default;
    left: 22px;
    width: 180px;
}
#mobile-toggle span{
    display: inline-block;
    font-family: 'ElegantIcons';
  speak: none;
  font-style: normal;
  font-weight: normal;
  font-variant: normal;
  text-transform: none;
  line-height: 1;
  -webkit-font-smoothing: antialiased;
}
.mobile-button span::before {
    content: "\62";
    color: #d3ac5b;
```

```
}
.main_block{
    position: relative;
    width: 100%;
    overflow: hidden;
}
#home{
    min-height: 840px;
}
/* slider */
#one.slider{
  width: 100%;
  height: 100%;
  overflow: hidden;
  position: absolute;
  left: 0;
  top: 0;
  z-index: 1;
}
.slider ul{
  width: 9999px;
  margin: 0;
  padding: 0;
  list-style: none;
  overflow: hidden;
  height: 100%;
}
.slider li{
  float: left;
  height: 100%;
}
.slider li{
    background-repeat: no-repeat;
    background-position: top center;
    background-size: cover;
    float: left;
    height: 100%;
}
#dark{
    position: relative;
    max-width: 1340px;
    margin: 56px auto 0;
    height: 730px;
    background: rgba(0, 0, 0, 0.7);
    z-index: 2;
}
.center{
    position: relative;
    max-width: 1340px;
    margin: 0 auto;
    z-index: 2;
}
#dark .topAddr{
    font-size: 10px;
    margin: 36px 0;
    letter-spacing: 3px;
}
#logo {
    position: relative;
    max-width: 691px;
    height: 158px;
    margin: 150px auto 0;
    background: url(../img/logo.png) no-repeat;
    background-size: contain;
    z-index: 2;
```

```
}
#dark h4{
    margin: 120px 0 20px;
    letter-spacing: 6px;
}
#dark .scroll-down {
    cursor: pointer;
}
#dark .scroll-down img {
    position: relative;
    top: 0;
    transition: all 300ms ease;
    -moz-transition: all 300ms ease;
    -ms-transition: all 300ms ease;
    -webkit-transition: all 300ms ease;
}
#dark .scroll-down:hover img {
    top: 15px;
}
#menu2{
    width: 100%;
}
#inner_menu{
    max-width: 1340px;
    margin: 0 auto;
    padding: 24px 0 22px;
    text-align: center;
}
#menu2 li:after {
    padding-left: 16px;
    color: #000;
}
#menu2 li a{
    color: #d3ac5b;
}
/*---------------------Design--------------------*/
#design{
    min-height: 1560px;
    background: #000;
}
#design .caption{
    margin: 150px 0 50px;
}
#two, #three, #four{
    display: inline-block;
    width: 100%;
    height: 840px;
}
#design p, #apartment p{
    max-width: 980px;
}
/*---------------------Apartment--------------------*/
#apartment p, #location p, #lifestyle p{
    color: #000;
}
#apartment .caption2{
    margin: 100px 0 50px;
}
#apartment .sentence{
    margin-bottom: 60px;
}
#full{
    width: 100%;
    height: auto;
    margin: 40px 0 0;
```

```
}
/*---------------------Location--------------------*/
#location .center, #lifestyle .center{
    margin-top: 130px;
}
#location .block{
    max-width: 415px;
    padding: 0 30px;
    float: left;
    text-align: left;
}
.block h2{
    margin: 0 0 20px;
    line-height: 40px;
}
#location .block2{
    max-width: 373px;
    margin-right: 12px;
    float: left;
}
#location .block2 .img1{
    margin-bottom: 12px;
}
#location .img3, #location .img5{
    float: right;
}
#location .img4, #location .img5{
    margin: 12px 0 16px;
}
/*-----------------Lifestyle-------------------------*/
#lifestyle .center{
    max-width: 1267px;
    padding: 0 10px;
}
#lifestyle img, #lifestyle .block2{
    float: left;
}
#lifestyle .block{
    float: right;
    text-align: left;
    max-width: 640px;
    padding: 0px 30px;
}
#lifestyle .block2{
    width: 100%;
    margin: 15px 0 17px;
}
#lifestyle .block2 .img2{
    margin-bottom: 17px;
}
#lifestyle .block2 .img3{
    margin-right: 14px;
}
#lifestyle .block2 .block3{
    max-width: 566px;
    float: left;
}
#lifestyle .block2 .img5{
    float: right;
    width: 687px;
    height: 718px;
}
/*-----------------The team-------------------------*/
#team, #enquire{
    background: #000;
```

```
}
#team .center{
    max-width: 1145px;
}
#team .caption3{
    margin: 130px 0 40px;
}
#team hr{
    background: #fcfffb;
    height: 4px;
    border: none;
    margin: 0;
}
#team .half{
    width: 510px;
    text-align: left;
    float: left;
    padding-right: 62px;
}
#team .half h2{
    margin: 10px 0 40px;
}
#team .half .partner{
    margin: 40px 0 100px;
}
/*-----------------Enquire-------------------------*/
#enquire .center{
    max-width: 1064px;
}
#enquire .caption4{
    margin: 110px 0 50px;
}
#enquire form{
    display: block;
    width: 545px;
    padding: 20px 0 0;
    margin-bottom: 50px;
    background: #fffbf5;
    float: left;
}
#enquire form input, #enquire form .dropdown{
    border: none;
    display: block;
    max-width: 470px;
    width: 100%;
    font-family: 'GothamMedium', sans-serif;
    font-size: 18px;
    font-weight: 400;
    height: 40px;
    color: #000;
    margin: 0 auto 14px;
    background: none;
    border-radius: 0;
    border-bottom: 2px solid #000;
    text-align: center;
    outline: none;
}
#enquire form .dropdown .selected{
    font-family: 'GothamMedium', sans-serif;
    font-size: 18px;
    font-weight: 400;
    outline: none;
}
#enquire .dropdown ul{
    padding: 0;
```

```
    margin: 0;
}
#enquire form .dropdown:hover {
    box-shadow: none;
}

#enquire form ::-webkit-input-placeholder {
    color: #000 !important;
    opacity: 1 !important;
}

#enquire form ::-moz-placeholder {
    color: #000 !important;
    opacity: 1 !important;
}

#enquire form :-ms-input-placeholder {
    color: #000 !important;
    opacity: 1 !important;
}

#enquire form :-moz-placeholder {
    color: #000 !important;
    opacity: 1 !important;
}

 #enquire form .input-error::-webkit-input-placeholder {
    color: red !important;
}

#enquire form .input-error::-moz-placeholder {
    color: red !important;
}

 #enquire form .input-error:-ms-input-placeholder {
    color: red !important;
}

 #enquire form .input-error:-moz-placeholder {
    color: red !important;
}

#enquire form label.error {
    display: none !important;
}

#enquire form button {
    width: 100%;
    border: none;
    background: #d3ac5b;
    font-family: 'GothamMedium', sans-serif;
    font-weight: 800;
    letter-spacing: 12.5px;
    text-align: center;
    height: 62px;
    font-size: 21px;
    color: #000;
    cursor: pointer;
    transition: all 300ms ease;
    -moz-transition: all 300ms ease;
    -ms-transition: all 300ms ease;
    -webkit-transition: all 300ms ease;
}

#enquire form button:hover {
```

```
    color: #fff;
}
#enquire .block{
    text-align: left;
    margin-left: 170px;
    float: left;
}
#enquire .caption5{
    margin: 0;
}
#enquire .caption6{
    margin: 0 0 40px;
}
#enquire .tel{
    font-size: 40px;
    margin-bottom: 40px;
}
#enquire .partner{
    margin-top: 46px;
}

/*-----------------Footer-------------------------*/
footer{
    height: 77px;
    background: #d3ac5b;
    text-align: left;
}
footer p{
    text-transform: uppercase;
    font-size: 10px;
    color: #000;
    display: block;
    float: left;
    margin: 50px 0 0 10px;
}
footer img{
    position: absolute;
    display: block;
    margin: 12px 0 0 -115px;
    left: 50%;
}
footer .notice2{
    float: right;
    margin: 50px 10px 0 0;
}
@media screen and (max-width: 1350px){
    #location .block, #location .block2, #location .img3{
        width: 30%;
        padding: 0 1.5%;
        margin: 0;
    }
    #location .block2 img,
    #lifestyle .block2 .img2{
        width: 100%;
        height: auto;
    }
    #location .img4{
        width: 90%;
    }
    #location .img5{
        display: none;
    }
    #lifestyle .img1{
        width: 50%;
    }
```

```
    #lifestyle .block{
        width: 46%;
        padding: 0 2%;
    }
    #lifestyle .block2 .block3,
    #lifestyle .block2 .img3,
    #lifestyle .block2 .img4,
    #lifestyle .block2 .img5{
        width: 48%;
        margin: 0 2% 0 0;
    }
    #lifestyle .block2 .img4,
    #lifestyle .block2 .img5{
        margin: 0 0 0 2%;
    }
    #lifestyle .block2 .img5{
        height: auto;
    }
    #team hr{
        width: 94%;
        margin: 0 auto;
    }
    #team .half{
        width: 46%;
        padding: 0 2%;
    }
}
@media screen and (max-width: 1100px){
    #menu li:first-child, #location .img3{
        display: none;
    }
    #menu{
        text-align: center;
    }
    #design .caption,
    #apartment .caption2,
    #enquire .caption4,
    #enquire .partner{
        margin: 50px 0 50px;
    }
    h2, p{
        padding-left: 20px;
        padding-right: 20px;
    }
    .slider{
        max-height: 600px;
    }
    #one.slider{
        max-height: none;
    }
    #design {
        min-height: 0;
    }
    #design p{
        margin-bottom: 20px;
    }
    #location .center, #lifestyle .center {
        margin-top: 100px;
    }
    #location .block, #location .block2{
        width: 46%;
        padding: 0 2%;
        margin: 0;
    }
    #enquire form{
```

```
        float: none;
        margin: 0 auto 50px;
    }
    #enquire .block{
        text-align: center;
        margin: 0 auto;
        float: none;
    }
}
@media screen and (max-width: 900px){
    #menu li:after{
        content: "";
        padding-left: 0;
    }
    #logo{
        margin: 50px auto 0;
    }
    #home{
        min-height: 0;
    }
    #one.slider{
        height: 600px;
    }
    #dark{
        height: 544px;
    }
    #dark h4 {
        margin: 66px 0 20px;
    }
    #menu2, #location .block2,
    #lifestyle .img1{
        display: none;
    }
    #location .center, #lifestyle .center {
        margin-top: 50px;
    }
    #location .block,
    #lifestyle .block{
        width: 96%;
        max-width: none;
        text-align: center;
        margin: 0 auto 10px;
    }
}
@media screen and (max-width: 700px){
    .new_nav{
        max-height: 48px;
        overflow: hidden;
    }
    .new_nav.show{
        max-height: 234px;
        transition: all 0.3s;
        -moz-transition: all 0.3s;
        -ms-transition: all 0.3s;
        -webkit-transition: all 0.3s;
    }
    .mobile-button{
        display: block;
    }
    #menu {
        padding: 0 22px 12px;
        text-align: left;
        margin-top: 48px!important;
        background: #fcfffb;
    }
```

```
    #menu li, #menu li:last-child, #menu.show li:first-child {
        width: 100%;
        float: none;
        padding: 5px 0;
        border-bottom: 1px dotted #967B4E;
    }
    #menu li a{
        font-family: 'GothamBook';
    }
    #logo,
    #enquire form,
    #enquire form input,
    #enquire form .dropdown{
        width: 90%;
    }
    .slider {
        max-height: 500px;
    }
    h2 {
        font-size: 36px;
    }
    p {
        font-size: 16px;
    }
    #location .center, #lifestyle .center {
        margin-top: 20px;
    }
    #team .caption3 {
        margin: 40px 0 30px;
    }
    #team .half.spec{
        width: 96%;
    }
    #team .half .partner {
        margin: 40px 0;
    }
    #design .caption,
    #apartment .caption2,
    #enquire .caption4,
    #enquire .partner {
        margin: 30px 0 50px;
    }
    footer img{
        width: 140px;
        margin: 20px 0 0 -70px;
    }
}
@media screen and (max-width: 480px){
    footer {
        height: 116px;
    }
    footer img{
        width: 230px;
        margin: 60px 0 0 -115px;
    }
}
```

jquery.bxslider.css

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Controls;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Media.Effects;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace WpfApp1
{
    /// <summary>
    /// Логика взаимодействия для MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public const double m = 0.6, H = 0.6, R = 1, A = 3, B = 2;
            public int koef=1, koefsize=1, kostyl=1;
        Polyline   prim;
        Polyline effect;
        Polyline patter;
        Polyline[,] orn = new Polyline[10,10];
        Point[] p = new Point [401];
        PointCollection pc;
        Color c;
        Brush b;
        Brush ib;
        System.Windows.Threading.DispatcherTimer timer;

        private System.Media.SoundPlayer mediaPlayer = new System.Media.SoundPlayer();

        System.Windows.Threading.DispatcherTimer animationTimer;
        int positionOfObject = 0;
        int positionOfDopka = 30;

        public MainWindow()
        {

            InitializeComponent();
            System.IO.Stream str = Properties.Resources.stillalive;
            mediaPlayer.Stream = str;
        }

        private void Primitive_grid_Loaded(object sender, RoutedEventArgs e)
        {
            Primitive_grid.Children.Remove(prim);
            Random r = new Random();
            prim = new Polyline();
            p = new Point[401];
            for (int i = 0; i < 401; i++)
            {
                p[i].X = Primitive_grid.ActualHeight * F(Math.PI * i / 40.0, 100) / 5.0 +
Primitive_grid.ActualWidth / 2;
                p[i].Y = Primitive_grid.ActualHeight * G(Math.PI * i / 40.0, 100) / 5.0 +
Primitive_grid.ActualHeight / 2;
            }
            pc = new PointCollection(p);
            prim.Points = pc;
            c = HSV(r.Next(360), 90, 75);
            b = new SolidColorBrush(c);
```

```
            ib = new SolidColorBrush(Color.FromRgb((byte)((255 - c.R) * 0.3), (byte)((255 - c.G) *
0.3), (byte)((255 - c.B) * 0.3)));
            prim.Stroke = b;
            prim.StrokeThickness = 4;
            Primitive_grid.Background = ib;
            Primitive_grid.Children.Add(prim);


        }

        public static double F(double t, int x0)
        {
            return (A / B) * R * Math.Cos(t / B) + (1 / B) * R * Math.Cos(A * t / B);
            //return ((R + m * R) * Math.Cos(m * t) + H * Math.Cos(t + m * t));
        }

        public static double G(double t, int y0)
        {
            return (A/B)*R*Math.Sin(t/B) - (1/B) * R * Math.Sin(A*t / B);
                //((R + m * R) * Math.Sin(m * t) + H * Math.Sin(t + m * t));
        }

        private void Button_Click(object sender, RoutedEventArgs e)
        {
            Primitive_grid.Children.Remove(prim);
            Random r = new Random();
            c = HSV(r.Next(360), 90, 75);
            b = new SolidColorBrush(c);
            ib = new SolidColorBrush(Color.FromRgb((byte)((255 - c.R) * 0.3), (byte)((255 - c.G) *
0.3), (byte)((255 - c.B) * 0.3)));
            prim.Stroke = b;
            Primitive_grid.Background = ib;
            Primitive_grid.Children.Add(prim);
        }

        private void Ornament_grid_Loaded(object sender, RoutedEventArgs e)
        {
            double a = Math.Min(Ornament_grid.ActualHeight, Ornament_grid.ActualWidth) / 10.0;
            for (int i = 0; i < 10; i++)
            {
                for (int j = 0; j < 10; j++)
                {
                    Ornament_grid.Children.Remove(orn[i, j]);
                    orn[i, j] = new Polyline();
                    p = new Point[401];
                    for (int k = 0; k < 401; k++)
                    {
                        p[k].Y = a * G(Math.PI * k / 20.0, 100) / 5.0 + (j - 4.5) * a +
Ornament_grid.ActualHeight / 2;
                        p[k].X = a * F(Math.PI * k / 20.0, 100) / 5.0 + (i - 4.5) * a +
Ornament_grid.ActualWidth / 2;
                    }
                    pc = new PointCollection(p);
                    orn[i, j].Points = pc;
                    orn[i, j].StrokeThickness = 6;
                    double b = Math.Sqrt(i * i + j * j);
                    Color c = HSV((int)(b * 28), 100, 25);
                    Brush bb = new SolidColorBrush(c);
                    orn[i, j].Stroke = bb;
                    Ornament_grid.Children.Add(orn[i, j]);

                    orn[i, j] = new Polyline();
                    p = new Point[401];
                    for (int k = 0; k < 401; k++)
                    {
```

```
                        p[k].Y = a * G(Math.PI * k / 20.0, 100) / 5.0 + (j - 4.5) * a +
Ornament_grid.ActualHeight / 2;
                        p[k].X = a * F(Math.PI * k / 20.0, 100) / 5.0 + (i - 4.5) * a +
Ornament_grid.ActualWidth / 2;
                    }
                    pc = new PointCollection(p);
                    orn[i, j].Points = pc;
                    orn[i, j].StrokeThickness = 1.5;
                    b = Math.Sqrt(i * i + j * j);
                    c = HSV((int)(b * 28), 100, 100);
                    bb = new SolidColorBrush(c);
                    orn[i, j].Stroke = bb;
                    Ornament_grid.Children.Add(orn[i, j]);


                }
            }
        }

        private void Window_Closing(object sender, System.ComponentModel.CancelEventArgs e)
        {
            if (timer != null)
                timer.Stop();
        }

        private void button_Click_1(object sender, RoutedEventArgs e)
        {
            image.Visibility = Visibility.Visible;
            button.Visibility = Visibility.Visible;
            label.Visibility = Visibility.Hidden;
        }

        private void button_Click_2(object sender, RoutedEventArgs e)
        {
            koef += 6;
        }

        private void Animation_grid_Loaded(object sender, RoutedEventArgs e)
        {
            if (animationTimer == null)
            {
                patter = new Polyline();
                Point[] pat = new Point[401];
                for (int i = 0; i < 401; i++)
                {
                    pat[i].X = Primitive_grid.ActualWidth  * F(Math.PI * i / 40.0, 100) / 5.0 +
Primitive_grid.ActualWidth / 2;
                    pat[i].Y = Primitive_grid.ActualHeight * G(Math.PI * i / 40.0, 100) / 5.0 +
Primitive_grid.ActualHeight / 2;
                }
                patter.Points = new PointCollection(pat);

                patter.Stroke = Brushes.Gray;
                patter.StrokeThickness = 4;

                Animation_grid.Children.Add(patter);
               // int k = 1;
                animationTimer = new System.Windows.Threading.DispatcherTimer();
                animationTimer.Interval = new TimeSpan(1000000 / 6);
                animationTimer.Tick += (ss, ee) =>
                {
                    if(koef > 1) koef--;

                    positionOfObject+=koef* kostyl;
                    positionOfDopka +=2*kostyl;
```

```
                    AnimatorButton.Margin = new Thickness(742 * F(Math.PI * (positionOfObject ) /
120.0, 100) / 5.0 + 742 / 2 - AnimatorButton.ActualWidth / 2,
                                                          742 * G(Math.PI * (positionOfObject ) /
120.0, 100) / 5.0 + 742 / 2 - AnimatorButton.ActualHeight , 0, 0);

                    image.Margin = new Thickness(742 * F(Math.PI * (positionOfDopka) / 120.0, 100) /
5.0 + 742 / 2 - image.ActualWidth / 2,
                                                          742 * G(Math.PI * (positionOfDopka) /
120.0, 100) / 5.0 + 742 / 2 - image.ActualHeight, 0, 0);

                    if (AnimatorButton.Margin.Top <= image.Margin.Top + 25 &&
AnimatorButton.Margin.Top >= image.Margin.Top - 25
                     && AnimatorButton.Margin.Left <= image.Margin.Left + 50 &&
AnimatorButton.Margin.Left >= image.Margin.Left - 50
                     && image.Visibility == Visibility.Visible)
                    {
                        label.Visibility = Visibility.Visible;
                        image.Visibility = Visibility.Hidden;
                        button.Visibility = Visibility.Hidden;
                        if (koefsize == 3) kostyl = -1;
                        if (koefsize == 1) kostyl = 1;
                        AnimatorButton.Width += 10*koefsize*kostyl;
                        koefsize+= kostyl;
                    }
                };
                animationTimer.Start();
            }
        }

        private void AnimatorButton_Click(object sender, RoutedEventArgs e)
        {
            //AnimatorButton.Width = 500;
            mediaPlayer.Play();
            // animationTimer.Stop();
            //koef += 3;
          //System.Threading.Thread.Sleep(1000);
            //AnimatorButton.Width = 75;
            //animationTimer.Start();


        }

        private void Effect_grid_Loaded(object sender, RoutedEventArgs e)
        {
            if(timer != null)
                timer.Stop();
            Effect_grid.Children.Remove(effect);
            Random r = new Random();
            effect = new Polyline();
            p = new Point[401];
            for (int i = 0; i < 401; i++)
            {
                p[i].X = Effect_grid.ActualHeight * F(Math.PI * i / 40.0, 100) / 5.0 +
Effect_grid.ActualWidth / 2;
                p[i].Y = Effect_grid.ActualHeight * G(Math.PI * i / 40.0, 100) / 5.0 +
Effect_grid.ActualHeight / 2;
            }
            pc = new PointCollection(p);
            effect.RenderTransformOrigin = new Point(0.5, 0.5);
            effect.Points = pc;
            //effect.Stroke = new SolidColorBrush(Colors.Red);
            effect.StrokeThickness = 15;
            BlurBitmapEffect myBlurEffect = new BlurBitmapEffect();
            myBlurEffect.Radius = 100;
```

```
        // Set the KernelType property of the blur. A KernalType of "Box"
        // creates less blur than the Gaussian kernal type.
        myBlurEffect.KernelType = KernelType.Box;

        effect.BitmapEffect = myBlurEffect;
        //-----------------------------------------------
        DoubleAnimation da = new DoubleAnimation();
        da.From = 0;
        da.To = 360;
        da.Duration = new Duration(TimeSpan.FromSeconds(5));
        da.RepeatBehavior = RepeatBehavior.Forever;
        RotateTransform rt = new RotateTransform();
        effect.RenderTransform = rt;
        rt.BeginAnimation(RotateTransform.AngleProperty, da);
        //ColorAnimation ca = new ColorAnimation();
        //ca.From = Colors.Red;

        //ca.To = Colors.Blue;

        Color c = HSV(0, 75, 100);
        Color c2 = HSV(0, 75, 100);
        timer = new System.Windows.Threading.DispatcherTimer();
        timer.Interval = TimeSpan.FromSeconds(0.05);
        int val;
        int val2;
        int currentStep = 0;
        timer.Tick += (ss, ee) =>
        {
            val = 3 * ++currentStep;
            val2 = (val <= 300) ? val + 59 : val - 300;
            c = HSV(val, 100, 100);
            c2 = HSV(val2, 100, 100);
            RadialGradientBrush rgb = new RadialGradientBrush();
            rgb.GradientStops.Add(new GradientStop(c2, 0.5));
            rgb.GradientStops.Add(new GradientStop(c, 1));
            effect.Stroke = rgb;
            if (currentStep >= 119)
                currentStep = 0;
        };
        timer.Start();


        //ca.Duration = new Duration(TimeSpan.FromSeconds(5));
        //effect.Stroke.BeginAnimation(SolidColorBrush.ColorProperty, ca);

        //-----------------------------------------------
        Effect_grid.Children.Add(effect);

    }

    public static Color HSV(int hue, int sat, int value)
    {
        double h = hue;
        double s = sat / 100.0;
        double v = value / 100.0;
        double r = 0, g = 0, b = 0;
        if (s == 0)
        {
            r = g = b = v;
        }
        else
        {
            double sector = h / 60.0;
            int sectorNumber = (int)Math.Floor(sector);
            double sectorPart = sector - sectorNumber;
```

```
                double p = v * (1 - s);
                double q = v * (1 - (s * sectorPart));
                double t = v * (1 - (s * (1 - sectorPart)));
                switch (sectorNumber)
                {
                    case 0: r = v; g = t; b = p; break;
                    case 1: r = q; g = v; b = p; break;
                    case 2: r = p; g = v; b = t; break;
                    case 3: r = p; g = q; b = v; break;
                    case 4: r = t; g = p; b = v; break;
                    case 5: r = v; g = p; b = q; break;
                }
            }
            return Color.FromRgb((byte)(r * 255), (byte)(g * 255), (byte)(b * 255));
        }
    }
}
```

hotspot-map.min.css

```
@media only screen {

  .hs-wrap {
    position: relative;
}
.hs-wrap * {
    display: none;
    box-sizing: content-box!important;
}
.hs-wrap img, .hs-wrap.hs-loaded * {
    display: block;
}
.hs-wrap.responsive, .hs-wrap.responsive img {
    width: 100%;
    /* margin-top: 10px; */
}
.hs-spot-object {
    position: absolute;
    cursor: pointer;
    z-index: 1;
}
.hs-spot-object.visible-tooltip {
    z-index: 9999;
}
.hs-spot.visible .hs-spot-shape {
    position: absolute;
    left: -1px;
    /* display: none; */
    display: none;
    top: -1px;
    background: #222;
    border-radius: 40px;
    -moz-border-radius: 40px;
    -webkit-border-radius: 40px;
    width: 100%;
    height: 100%;
    -ms-filter: "alpha(Opacity=15)";
    filter: alpha(opacity=15);
    -moz-opacity: .15;
    -khtml-opacity: .15;
    opacity: .35;
    z-index: 0;
    border: 1px solid #f8f7ee;
    transform: scale3d(1, 1, 1);
```

```
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot.visible .hs-spot-shape-inner {
    background: #da0606;
    position: absolute;
    left: 50%;
    top: 50%;
    display: none;
    width: 18px;
    height: 18px;
    margin: -9px 0 0 -9px;
    z-index: 1;
    border-radius: 30px;
    -moz-border-radius: 30px;
    -webkit-border-radius: 30px;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot.visible:hover .hs-spot-shape-inner {
    transform: scale3d(1.4, 1.4, 1.4);
    -moz-transform: scale3d(1.4, 1.4, 1.4);
    -webkit-transform: scale3d(1.4, 1.4, 1.4);
}
.hs-rect {
    position: absolute;
    left: 0;
    top: 0;
    cursor: pointer;
    z-index: 99;
    border: none;
}
.hs-rect.visible .hs-spot-shape {
    position: absolute;
    left: -3px;
    top: -3px;
    z-index: 1;
    width: 100%;
    height: 100%;
    background: 0 0;
    border: 3px solid #da0606;
    border-radius: 6px;
    -moz-border-radius: 6px;
    -webkit-border-radius: 6px;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-rect.visible .hs-spot-shape-inner {
    position: absolute;
    z-index: 0;
    width: 100%;
    height: 100%;
    background: 0 0;
    border: 8px solid #f8f7ee;
```

```
    border-radius: 8px;
    -moz-border-radius: 8px;
    -webkit-border-radius: 8px;
    left: -8px;
    top: -8px;
    -ms-filter: "alpha(Opacity=15)";
    filter: alpha(opacity=15);
    -moz-opacity: .15;
    -khtml-opacity: .15;
    opacity: .15;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-rect.visible:hover .hs-spot-shape {
    border-width: 5px;
    left: -5px;
    top: -5px;
}
.hs-rect.visible:hover .hs-spot-shape-inner {
    border-width: 7px;
    left: -7px;
    top: -7px;
}
.hs-spot-object .hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: 0;
}
.hs-tooltip-wrap {
    position: absolute;
}
.hs-tooltip {
    display: block;
    background: #f8f7ee;
    color: #555;
    /* border: 10px solid #f00; */
    -webkit-box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25); -moz-box-shadow: 2px 2px 2px 0px
rgba(0,0,0,0.25); box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25);
    font-family: "kepler-std",serif;
    font-size: 14px;
    text-transform: uppercase;
    padding: 10px;
    position: relative;
    min-width: 150px;
    min-height: 18px;
    border-radius: 3px;
    -moz-border-radius: 3px;
    -webkit-border-radius: 3px;
}

.hs-tooltip img{ margin-top:5px; border-top:2px solid #d8ebd2; border-bottom:4px solid #d8ebd2; }


.hs-wrap.click .hs-spot-object.left .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-object.left
.hs-spot-tooltip-outer {
    position: absolute;
    top: 0;
    left: -30px;
}
.hs-spot-object.left .hs-tooltip-wrap {
    right: 100%;
```

```
    top: 0;
    padding-right: 16px;
}
.hs-spot.left .hs-tooltip-wrap {
    top: -12px;
}
.hs-spot-object.left .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    width: 0;
    height: 0;
    right: -8px;
    top: 8px;
    border-top: 8px solid transparent;
    border-bottom: 8px solid transparent;
    border-left: 8px solid #f8f7ee;
}
.hs-spot.left .hs-tooltip:before {
    top: 16px;
}
.hs-wrap.click .hs-spot-object.top .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-object.top
.hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: -30px;
}
.hs-spot-object.top .hs-tooltip-wrap {
    bottom: 100%;
    left: 0;
    padding-bottom: 16px;
}
.hs-spot.top .hs-tooltip-wrap {
    left: -1px;
}
.hs-spot-object.top .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    left: 8px;
    bottom: -8px;
    width: 0;
    height: 0;
    border-left: 8px solid transparent;
    border-right: 8px solid transparent;
    border-top: 8px solid #f8f7ee;
}
.hs-wrap.click .hs-spot-object.right .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-
object.right .hs-spot-tooltip-outer {
    position: absolute;
    top: 0;
    left: 30px;
}
.hs-spot-object.right .hs-tooltip-wrap {
    left: 100%;
    top: 0;
    padding-left: 16px;
}
.hs-spot.right .hs-tooltip-wrap {
    top: -12px;
    /* border: 10px solid #f00; */
}
.hs-spot-object.right .hs-tooltip:before {
    position: absolute;
    content: "";
```

```
    display: block;
    left: -8px;
    top: 8px;
    width: 0;
    height: 0;
    border-top: 8px solid transparent;
    border-bottom: 8px solid transparent;
    border-right: 8px solid #f8f7ee;
}
.hs-spot.right .hs-tooltip:before {
    top: 16px;
}
.hs-wrap.click .hs-spot-object.bottom .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-
object.bottom .hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: 30px;
}
.hs-spot-object.bottom .hs-tooltip-wrap {
    top: 100%;
    left: 0;
    padding-top: 16px;
}
.hs-spot.bottom .hs-tooltip-wrap {
    left: -1px;
}
.hs-spot-object.bottom .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    top: -8px;
    left: 8px;
    width: 0;
    height: 0;
    border-left: 8px solid transparent;
    border-right: 8px solid transparent;
    border-bottom: 8px solid #f8f7ee;
}
.hs-tooltip-buffer {
    background: #00f;
    position: absolute;
    left: 0;
    /* border: 10px solid #f00; */
    top: 0;
    width: 100%;
    height: 100%;
    z-index: 999;
    -ms-filter: "alpha(Opacity=0)";
    filter: alpha(opacity=0);
    -moz-opacity: 0;
    -khtml-opacity: 0;
    opacity: 0;
}
.hs-spot-object.bottom .hs-tooltip-buffer {
    top: 100%;
    height: 16px;
}
.hs-spot-object.top .hs-tooltip-buffer {
    top: auto;
    bottom: 100%;
    height: 16px;
}
.hs-spot-object.left .hs-tooltip-buffer {
    right: 100%;
    left: auto;
```

```
    width: 16px;
}
.hs-spot-object.right .hs-tooltip-buffer {
    left: 100%;
    width: 16px;
}
.hs-tooltip h1 {
    font: 14px/14px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip h2 {
    font: 12px/12px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip h3 {
    font: 11px/11px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip p {
    font: 11px/18px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
}
.hs-tooltip :last-child {
    margin-bottom: 0;
}
.hs-tooltip a {
    color: #fff!important;
    text-decoration: underline!important;
}
.hs-tooltip a:hover {
    text-decoration: none!important;
}
.hs-wrap.always .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=100)";
    filter: alpha(opacity=100);
    -moz-opacity: 1;
    -khtml-opacity: 1;
    opacity: 1;
    left: 0!important;
    top: 0!important;
    width: 100%!important;
    height: 100%!important;
}
.hs-wrap.always .hs-tooltip, .hs-wrap.always .hs-tooltip-wrap {
    display: block;
}
.hs-spot-object .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=0)";
    filter: alpha(opacity=0);
    -moz-opacity: 0;
    -khtml-opacity: 0;
    opacity: 0;
    width: 0!important;
    height: 0!important;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot-object .hs-tooltip, .hs-spot-object .hs-tooltip-wrap {
```

```
    display: none;
}
.hs-spot-object.visible-tooltip .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=100)";
    filter: alpha(opacity=100);
    -moz-opacity: 1;
    -khtml-opacity: 1;
    opacity: 1;
    left: 0!important;
    top: 0!important;
    width: 100%!important;
    height: 100%!important;
}
.hs-spot-object.visible-tooltip .hs-tooltip, .hs-spot-object.visible-tooltip .hs-tooltip-wrap {
    display: block;
}
}


@media only screen and (min-width: 40.063em) {

  .hs-wrap {
    position: relative;
}
.hs-wrap * {
    display: none;
    box-sizing: content-box!important;
}
.hs-wrap img, .hs-wrap.hs-loaded * {
    display: block;
}
.hs-wrap.responsive, .hs-wrap.responsive img {
    width: 100%;
    /* margin-top: 10px; */
}
.hs-spot-object {
    position: absolute;
    cursor: pointer;
    z-index: 1;
}
.hs-spot-object.visible-tooltip {
    z-index: 9999;
}
.hs-spot.visible .hs-spot-shape {
    position: absolute;
    left: -1px;
    top: -1px;
    background: #222;
    display: block;
    border-radius: 40px;
    -moz-border-radius: 40px;
    -webkit-border-radius: 40px;
    width: 100%;
    height: 100%;
    -ms-filter: "alpha(Opacity=15)";
    filter: alpha(opacity=15);
    -moz-opacity: .15;
    -khtml-opacity: .15;
    opacity: .35;
    z-index: 0;
    border: 1px solid #f8f7ee;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
```

```
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot.visible .hs-spot-shape-inner {
    background: #da0606;
    position: absolute;
    left: 50%;
    top: 50%;
    width: 18px;
    display: block;
    height: 18px;
    margin: -9px 0 0 -9px;
    z-index: 1;
    border-radius: 30px;
    -moz-border-radius: 30px;
    -webkit-border-radius: 30px;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot.visible:hover .hs-spot-shape-inner {
    transform: scale3d(1.4, 1.4, 1.4);
    -moz-transform: scale3d(1.4, 1.4, 1.4);
    -webkit-transform: scale3d(1.4, 1.4, 1.4);
}
.hs-rect {
    position: absolute;
    left: 0;
    top: 0;
    cursor: pointer;
    z-index: 99;
    border: none;
}
.hs-rect.visible .hs-spot-shape {
    position: absolute;
    left: -3px;
    top: -3px;
    z-index: 1;
    width: 100%;
    height: 100%;
    background: 0 0;
    border: 3px solid #da0606;
    border-radius: 6px;
    -moz-border-radius: 6px;
    -webkit-border-radius: 6px;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-rect.visible .hs-spot-shape-inner {
    position: absolute;
    z-index: 0;
    width: 100%;
    height: 100%;
    background: 0 0;
    border: 8px solid #f8f7ee;
    border-radius: 8px;
    -moz-border-radius: 8px;
    -webkit-border-radius: 8px;
```

```
    left: -8px;
    top: -8px;
    -ms-filter: "alpha(Opacity=15)";
    filter: alpha(opacity=15);
    -moz-opacity: .15;
    -khtml-opacity: .15;
    opacity: .15;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-rect.visible:hover .hs-spot-shape {
    border-width: 5px;
    left: -5px;
    top: -5px;
}
.hs-rect.visible:hover .hs-spot-shape-inner {
    border-width: 7px;
    left: -7px;
    top: -7px;
}
.hs-spot-object .hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: 0;
}
.hs-tooltip-wrap {
    position: absolute;
}
.hs-tooltip {
    display: block;
    background: #fff;
    color: #444;
    /* border: 10px solid #f00; */
    -webkit-box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25);
    -moz-box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25);
    box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25);
    font-family: 'BrownStd-Regular';
    font-size: 12px;
    text-transform: uppercase;
    padding: 10px;
    position: relative;
    min-width: 150px;
    min-height: 18px;
    border-radius: 3px;
    -moz-border-radius: 3px;
    -webkit-border-radius: 3px;
}

.hs-tooltip img{ margin-top:5px; border-top:2px solid #d8ebd2; border-bottom:4px solid #d8ebd2; }


.hs-wrap.click .hs-spot-object.left .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-object.left
.hs-spot-tooltip-outer {
    position: absolute;
    top: 0;
    left: -30px;
}
.hs-spot-object.left .hs-tooltip-wrap {
    right: 100%;
    top: 0;
    padding-right: 16px;
```

```
}
.hs-spot.left .hs-tooltip-wrap {
    top: -12px;
}
.hs-spot-object.left .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    width: 0;
    height: 0;
    right: -8px;
    top: 8px;
    border-top: 8px solid transparent;
    border-bottom: 8px solid transparent;
    border-left: 8px solid #f8f7ee;
}
.hs-spot.left .hs-tooltip:before {
    top: 16px;
}
.hs-wrap.click .hs-spot-object.top .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-object.top
.hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: -30px;
}
.hs-spot-object.top .hs-tooltip-wrap {
    bottom: 100%;
    left: 0;
    padding-bottom: 16px;
}
.hs-spot.top .hs-tooltip-wrap {
    left: -1px;
}
.hs-spot-object.top .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    left: 8px;
    bottom: -8px;
    width: 0;
    height: 0;
    border-left: 8px solid transparent;
    border-right: 8px solid transparent;
    border-top: 8px solid #f8f7ee;
}
.hs-wrap.click .hs-spot-object.right .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-
object.right .hs-spot-tooltip-outer {
    position: absolute;
    top: 0;
    left: 30px;
}
.hs-spot-object.right .hs-tooltip-wrap {
    left: 100%;
    top: 0;
    padding-left: 16px;
}
.hs-spot.right .hs-tooltip-wrap {
    top: -12px;
    /* border: 10px solid #f00; */
}
.hs-spot-object.right .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    left: -8px;
```

```
    top: 8px;
    width: 0;
    height: 0;
    border-top: 8px solid transparent;
    border-bottom: 8px solid transparent;
    border-right: 8px solid #f8f7ee;
}
.hs-spot.right .hs-tooltip:before {
    top: 16px;
}
.hs-wrap.click .hs-spot-object.bottom .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-
object.bottom .hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: 30px;
}
.hs-spot-object.bottom .hs-tooltip-wrap {
    top: 100%;
    left: 0;
    padding-top: 16px;
}
.hs-spot.bottom .hs-tooltip-wrap {
    left: -1px;
}
.hs-spot-object.bottom .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    top: -8px;
    left: 8px;
    width: 0;
    height: 0;
    border-left: 8px solid transparent;
    border-right: 8px solid transparent;
    border-bottom: 8px solid #f8f7ee;
}
.hs-tooltip-buffer {
    background: #00f;
    position: absolute;
    left: 0;
    /* border: 10px solid #f00; */
    top: 0;
    width: 100%;
    height: 100%;
    z-index: 999;
    -ms-filter: "alpha(Opacity=0)";
    filter: alpha(opacity=0);
    -moz-opacity: 0;
    -khtml-opacity: 0;
    opacity: 0;
}
.hs-spot-object.bottom .hs-tooltip-buffer {
    top: 100%;
    height: 16px;
}
.hs-spot-object.top .hs-tooltip-buffer {
    top: auto;
    bottom: 100%;
    height: 16px;
}
.hs-spot-object.left .hs-tooltip-buffer {
    right: 100%;
    left: auto;
    width: 16px;
}
```

```
.hs-spot-object.right .hs-tooltip-buffer {
    left: 100%;
    width: 16px;
}
.hs-tooltip h1 {
    font: 14px/14px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip h2 {
    font: 12px/12px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip h3 {
    font: 11px/11px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip p {
    font: 11px/18px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
}
.hs-tooltip :last-child {
    margin-bottom: 0;
}
.hs-tooltip a {
    color: #fff!important;
    text-decoration: underline!important;
}
.hs-tooltip a:hover {
    text-decoration: none!important;
}
.hs-wrap.always .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=100)";
    filter: alpha(opacity=100);
    -moz-opacity: 1;
    -khtml-opacity: 1;
    opacity: 1;
    left: 0!important;
    top: 0!important;
    width: 100%!important;
    height: 100%!important;
}
.hs-wrap.always .hs-tooltip, .hs-wrap.always .hs-tooltip-wrap {
    display: block;
}
.hs-spot-object .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=0)";
    filter: alpha(opacity=0);
    -moz-opacity: 0;
    -khtml-opacity: 0;
    opacity: 0;
    width: 0!important;
    height: 0!important;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot-object .hs-tooltip, .hs-spot-object .hs-tooltip-wrap {
    display: none;
}
```

```
.hs-spot-object.visible-tooltip .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=100)";
    filter: alpha(opacity=100);
    -moz-opacity: 1;
    -khtml-opacity: 1;
    opacity: 1;
    left: 0!important;
    top: 0!important;
    width: 100%!important;
    height: 100%!important;
}
.hs-spot-object.visible-tooltip .hs-tooltip, .hs-spot-object.visible-tooltip .hs-tooltip-wrap {
    display: block;
}
}


@media only screen and (min-width: 64.063em) {

  .hs-wrap {
    position: relative;
}
.hs-wrap * {
    display: none;
    box-sizing: content-box!important;
}
.hs-wrap img, .hs-wrap.hs-loaded * {
    display: block;
}
.hs-wrap.responsive, .hs-wrap.responsive img {
    width: 100%;
    /* margin-top: 10px; */
}
.hs-spot-object {
    position: absolute;
    cursor: pointer;
    z-index: 1;
}
.hs-spot-object.visible-tooltip {
    z-index: 9999;
}
.hs-spot.visible .hs-spot-shape {
    position: absolute;
    left: -1px;
    top: -1px;
    background: #222;
    border-radius: 40px;
    -moz-border-radius: 40px;
    -webkit-border-radius: 40px;
    width: 100%;
    height: 100%;
    -ms-filter: "alpha(Opacity=15)";
    filter: alpha(opacity=15);
    -moz-opacity: .15;
    -khtml-opacity: .15;
    opacity: .35;
    z-index: 0;
    border: 1px solid #f8f7ee;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
```

```
.hs-spot.visible .hs-spot-shape-inner {
    background: #da0606;
    position: absolute;
    left: 50%;
    top: 50%;
    width: 18px;
    height: 18px;
    margin: -9px 0 0 -9px;
    z-index: 1;
    border-radius: 30px;
    -moz-border-radius: 30px;
    -webkit-border-radius: 30px;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot.visible:hover .hs-spot-shape-inner {
    transform: scale3d(1.4, 1.4, 1.4);
    -moz-transform: scale3d(1.4, 1.4, 1.4);
    -webkit-transform: scale3d(1.4, 1.4, 1.4);
}
.hs-rect {
    position: absolute;
    left: 0;
    top: 0;
    cursor: pointer;
    z-index: 99;
    border: none;
}
.hs-rect.visible .hs-spot-shape {
    position: absolute;
    left: -3px;
    top: -3px;
    z-index: 1;
    width: 100%;
    height: 100%;
    background: 0 0;
    border: 3px solid #da0606;
    border-radius: 6px;
    -moz-border-radius: 6px;
    -webkit-border-radius: 6px;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-rect.visible .hs-spot-shape-inner {
    position: absolute;
    z-index: 0;
    width: 100%;
    height: 100%;
    background: 0 0;
    border: 8px solid #f8f7ee;
    border-radius: 8px;
    -moz-border-radius: 8px;
    -webkit-border-radius: 8px;
    left: -8px;
    top: -8px;
    -ms-filter: "alpha(Opacity=15)";
    filter: alpha(opacity=15);
```

```
    -moz-opacity: .15;
    -khtml-opacity: .15;
    opacity: .15;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-rect.visible:hover .hs-spot-shape {
    border-width: 5px;
    left: -5px;
    top: -5px;
}
.hs-rect.visible:hover .hs-spot-shape-inner {
    border-width: 7px;
    left: -7px;
    top: -7px;
}
.hs-spot-object .hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: 0;
}
.hs-tooltip-wrap {
    position: absolute;
}
.hs-tooltip {
    display: block;
    background: #fff;
    color: #444;
    /* border: 10px solid #f00; */
    -webkit-box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25);
    -moz-box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25);
    box-shadow: 2px 2px 2px 0px rgba(0,0,0,0.25);
    font-family: 'BrownStd-Regular';
    font-size: 13px;
    text-transform: uppercase;
    padding: 10px;
    position: relative;
    min-width: 150px;
    min-height: 18px;
    border-radius: 3px;
    -moz-border-radius: 3px;
    -webkit-border-radius: 3px;
}

.hs-tooltip img{ margin-top:5px; border-top:2px solid #d8ebd2; border-bottom:4px solid #d8ebd2; }


.hs-wrap.click .hs-spot-object.left .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-object.left
.hs-spot-tooltip-outer {
    position: absolute;
    top: 0;
    left: -30px;
}
.hs-spot-object.left .hs-tooltip-wrap {
    right: 100%;
    top: 0;
    padding-right: 16px;
}
.hs-spot.left .hs-tooltip-wrap {
    top: -12px;
}
```

```
.hs-spot-object.left .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    width: 0;
    height: 0;
    right: -8px;
    top: 8px;
    border-top: 8px solid transparent;
    border-bottom: 8px solid transparent;
    border-left: 8px solid #f8f7ee;
}
.hs-spot.left .hs-tooltip:before {
    top: 16px;
}
.hs-wrap.click .hs-spot-object.top .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-object.top
.hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: -30px;
}
.hs-spot-object.top .hs-tooltip-wrap {
    bottom: 100%;
    left: 0;
    padding-bottom: 16px;
}
.hs-spot.top .hs-tooltip-wrap {
    left: -1px;
}
.hs-spot-object.top .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    left: 8px;
    bottom: -8px;
    width: 0;
    height: 0;
    border-left: 8px solid transparent;
    border-right: 8px solid transparent;
    border-top: 8px solid #f8f7ee;
}
.hs-wrap.click .hs-spot-object.right .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-
object.right .hs-spot-tooltip-outer {
    position: absolute;
    top: 0;
    left: 30px;
}
.hs-spot-object.right .hs-tooltip-wrap {
    left: 100%;
    top: 0;
    padding-left: 16px;
}
.hs-spot.right .hs-tooltip-wrap {
    top: -12px;
    /* border: 10px solid #f00; */
}
.hs-spot-object.right .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    left: -8px;
    top: 8px;
    width: 0;
    height: 0;
    border-top: 8px solid transparent;
```

```
    border-bottom: 8px solid transparent;
    border-right: 8px solid #f8f7ee;
}
.hs-spot.right .hs-tooltip:before {
    top: 16px;
}
.hs-wrap.click .hs-spot-object.bottom .hs-spot-tooltip-outer, .hs-wrap.mouseover .hs-spot-
object.bottom .hs-spot-tooltip-outer {
    position: absolute;
    left: 0;
    top: 30px;
}
.hs-spot-object.bottom .hs-tooltip-wrap {
    top: 100%;
    left: 0;
    padding-top: 16px;
}
.hs-spot.bottom .hs-tooltip-wrap {
    left: -1px;
}
.hs-spot-object.bottom .hs-tooltip:before {
    position: absolute;
    content: "";
    display: block;
    top: -8px;
    left: 8px;
    width: 0;
    height: 0;
    border-left: 8px solid transparent;
    border-right: 8px solid transparent;
    border-bottom: 8px solid #f8f7ee;
}
.hs-tooltip-buffer {
    background: #00f;
    position: absolute;
    left: 0;
    /* border: 10px solid #f00; */
    top: 0;
    width: 100%;
    height: 100%;
    z-index: 999;
    -ms-filter: "alpha(Opacity=0)";
    filter: alpha(opacity=0);
    -moz-opacity: 0;
    -khtml-opacity: 0;
    opacity: 0;
}
.hs-spot-object.bottom .hs-tooltip-buffer {
    top: 100%;
    height: 16px;
}
.hs-spot-object.top .hs-tooltip-buffer {
    top: auto;
    bottom: 100%;
    height: 16px;
}
.hs-spot-object.left .hs-tooltip-buffer {
    right: 100%;
    left: auto;
    width: 16px;
}
.hs-spot-object.right .hs-tooltip-buffer {
    left: 100%;
    width: 16px;
}
```

```
.hs-tooltip h1 {
    font: 14px/14px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip h2 {
    font: 12px/12px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip h3 {
    font: 11px/11px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
    font-weight: 700;
}
.hs-tooltip p {
    font: 11px/18px helvetica, tahoma, sans-serif;
    margin-bottom: 10px;
}
.hs-tooltip :last-child {
    margin-bottom: 0;
}
.hs-tooltip a {
    color: #fff!important;
    text-decoration: underline!important;
}
.hs-tooltip a:hover {
    text-decoration: none!important;
}
.hs-wrap.always .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=100)";
    filter: alpha(opacity=100);
    -moz-opacity: 1;
    -khtml-opacity: 1;
    opacity: 1;
    left: 0!important;
    top: 0!important;
    width: 100%!important;
    height: 100%!important;
}
.hs-wrap.always .hs-tooltip, .hs-wrap.always .hs-tooltip-wrap {
    display: block;
}
.hs-spot-object .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=0)";
    filter: alpha(opacity=0);
    -moz-opacity: 0;
    -khtml-opacity: 0;
    opacity: 0;
    width: 0!important;
    height: 0!important;
    transform: scale3d(1, 1, 1);
    -moz-transform: scale3d(1, 1, 1);
    -webkit-transform: scale3d(1, 1, 1);
    transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -moz-transition: all .25s cubic-bezier(.55, 0, .1, 1);
    -webkit-transition: all .25s cubic-bezier(.55, 0, .1, 1);
}
.hs-spot-object .hs-tooltip, .hs-spot-object .hs-tooltip-wrap {
    display: block;
}
.hs-spot-object.visible-tooltip .hs-spot-tooltip-outer {
    -ms-filter: "alpha(Opacity=100)";
    filter: alpha(opacity=100);
    -moz-opacity: 1;
```

```
    -khtml-opacity: 1;
    opacity: 1;
    left: 0!important;
    top: 0!important;
    width: 100%!important;
    height: 100%!important;
}
.hs-spot-object.visible-tooltip .hs-tooltip, .hs-spot-object.visible-tooltip .hs-tooltip-wrap {
    display: block;
}
}
```

easydropdown.css

```
/* --- EASYDROPDOWN DEFAULT THEME --- */

/* PREFIXED CSS */

.dropdown,
.dropdown div,
.dropdown li,
.dropdown div::after{
  -webkit-transition: all 150ms ease-in-out;
  -moz-transition: all 150ms ease-in-out;
  -ms-transition: all 150ms ease-in-out;
  transition: all 150ms ease-in-out;
}

.dropdown .selected::after,
.dropdown.scrollable div::after{
  -webkit-pointer-events: none;
  -moz-pointer-events: none;
  -ms-pointer-events: none;
  pointer-events: none;
}

/* WRAPPER */

.dropdown{
  position: relative;
  width: 160px;
  border: 1px solid #ccc;
  cursor: pointer;
  background: #fff;

  border-radius: 3px;

  -webkit-user-select: none;
  -moz-user-select: none;
  user-select: none;
}

.dropdown.open{
  z-index: 2;
}

.dropdown:hover{
  box-shadow: 0 0 5px rgba(0,0,0,.15);
}

.dropdown.focus{
  box-shadow: 0 0 5px rgba(51,102,248,.4);
}
```

```
/* CARAT */

.dropdown .carat{
 position: absolute;
 right: 12px;
 top: 50%;
 margin-top: -4px;
 border: 6px solid transparent;
 border-top: 8px solid #000;
}

.dropdown.open .carat{
 margin-top: -10px;
 border-top: 6px solid transparent;
 border-bottom: 8px solid #000;
}

.dropdown.disabled .carat{
 border-top-color: #999;
}

/* OLD SELECT (HIDDEN) */

.dropdown .old{
 position: absolute;
 left: 0;
 top: 0;
 height: 0;
 width: 0;
 overflow: hidden;
}

.dropdown select{
 position: absolute;
 left: 0px;
 top: 0px;
}

.dropdown.touch .old{
 width: 100%;
 height: 100%;
}

.dropdown.touch select{
 width: 100%;
 height: 100%;
 opacity: 0;
}

/* SELECTED FEEDBACK ITEM */

.dropdown .selected,
.dropdown li{
 display: block;
 font-size: 18px;
 line-height: 1;
 color: #000;
 padding: 5px 12px;
 overflow: hidden;
 white-space: nowrap;
}

.dropdown.disabled .selected{
 color: #999;
```

```
}

.dropdown .selected::after{
 content: '';
 position: absolute;
 right: 0;
 top: 0;
 bottom: 0;
 width: 60px;

 border-radius: 0 2px 2px 0;
 box-shadow: inset -55px 0 25px -20px #fff;
}

/* DROP DOWN WRAPPER */

.dropdown div{
 position: absolute;
 height: 0;
 left: -1px;
 right: -1px;
 top: 100%;
 margin-top: -1px;
 background: #fff;
 border: 1px solid #ccc;
 border-top: 1px solid #eee;
 border-radius: 0 0 3px 3px;
 overflow: hidden;
 opacity: 0;
}

/* Height is adjusted by JS on open */

.dropdown.open div{
 opacity: 1;
 z-index: 2;
}

/* FADE OVERLAY FOR SCROLLING LISTS */

.dropdown.scrollable div::after{
 content: '';
 position: absolute;
 left: 0;
 right: 0;
 bottom: 0;
 height: 50px;

 box-shadow: inset 0 -50px 30px -35px #fff;
}

.dropdown.scrollable.bottom div::after{
 opacity: 0;
}

/* DROP DOWN LIST */

.dropdown ul{
 position: absolute;
 left: 0;
 top: 0;
 height: 100%;
 width: 100%;
 list-style: none;
 overflow: hidden;
```

```
}

.dropdown.scrollable.open ul{
 overflow-y: auto;
}

/* DROP DOWN LIST ITEMS */

.dropdown li{
 list-style: none;
 padding: 8px 12px;
}

/* .focus class is also added on hover */

.dropdown li.focus{
 background: #d3ac5b;
 position: relative;
 z-index: 3;
 color: #fff;
}

.dropdown li.active{
 font-weight: 700;
}


.dropdown .selected::after{
 content: '';
 position: absolute;
 right: 0;
 top: 0;
 bottom: 0;
 width: 60px;
 border-radius: 0 5px 5px 0;
 box-shadow: inset -55px 0 25px -20px #fffbf5;
}

.dropdown:hover .selected::after,
.dropdown.focus .selected::after{
 box-shadow: inset -55px 0 25px -20px #fffbf5;
}

.dropdown div{
 bottom: 100%;
 top: auto;
}
```